PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-050554
(43)Date of publication of application : 20.02.1996

(51)Int.Cl. G06F 11/22
G06F 9/38

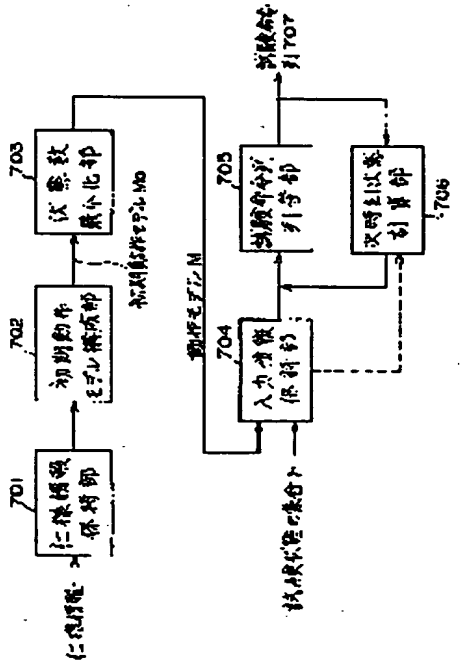
(21)Application number : 07-060487
(22)Date of filing : 20.03.1995

(71)Applicant : FUJITSU LTD
(72)Inventor : IWASHITA HIROAKI
FURUWATARI SATOSHI
NAKADA TSUNEO
HIROSE FUMIYASU

(30)Priority
Priority number : 06116705 Priority date : 30.05.1994 Priority country : JP

(54) METHOD AND DEVICE FOR AUTOMATICALLY GENERATING OPERATION MODEL OF PROCESSOR AND TEST INSTRUCTION SEQUENCE FOR LOGIC VERIFICATION

(57)Abstract:
PURPOSE: To automatically generate a high-quality test instruction sequence in a short time for verifying the operation logic of a processor equipped with a pipelined processing function.
CONSTITUTION: A part 701 holds specification information concerning the pipelined configuration of the processor and an instruction to be executed by the processor. A part 702 constitutes an initial operation model M0 concerning a pipeline from the specification information. A part 703 constitutes an operation model M by minimizing the number of states of the initial operation model M0. A part 705 successively presents a test instruction sequence 707 corresponding to the process to transit from a prescribed input state to any test state contained in a set H of test states without competing the state of the operation model M based on the operation model M held in the part 704 and the set H of test states. A part 706 calculates the next time state of a state to be next transit after the state of the operation model M is turned to the test state corresponding to the test instruction sequence 707, and that next time state is inputted to a successive test instruction sequence presenting means 705 as the new input state.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－50554

(43)公開日　平成８年(1996)２月20日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　11/22	３１０　Ｕ			
9/38	３８０　Ｃ			

審査請求　未請求　請求項の数18　ＯＬ　（全 46 頁）

(21)出願番号　特願平7－60487

(22)出願日　平成７年(1995)３月20日

(31)優先権主張番号　特願平6－116705

(32)優先日　平６(1994)５月30日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地

(72)発明者　岩下　洋哲
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(72)発明者　古渡　聡
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(72)発明者　中田　恒夫
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74)代理人　弁理士　大菅　義之　（外１名）

最終頁に続く

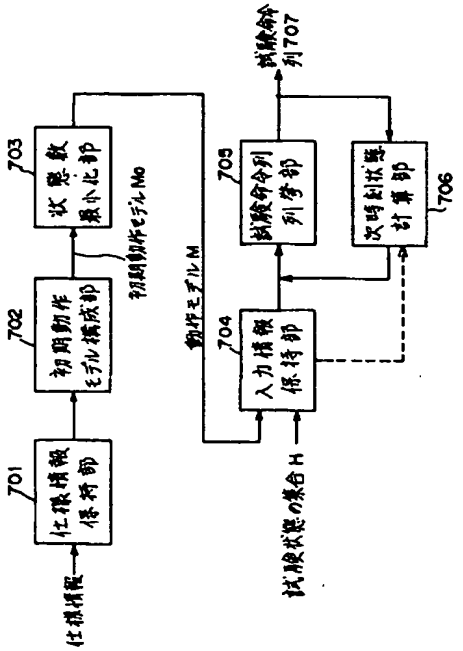
(54)【発明の名称】　プロセッサの動作モデルと論理検証用試験命令列の自動生成方法及び装置

(57)【要約】

【目的】　パイプライン処理機能を有するプロセッサの動作論理を検証するため質の高い試験命令列を自動的かつ短時間に生成可能とすることを目的とする。

【構成】　７０１の部分は、プロセッサのパイプライン構成及びプロセッサが実行する命令に関する仕様情報を保持する。７０２の部分は、仕様情報から、パイプラインに関する初期動作モデルM_0を構成する。７０３の部分は、初期動作モデルM_0の状態の数を最小化することにより、動作モデルMを構成する。７０５の部分は、７０４の部分に保持された動作モデルMと試験状態の集合Hとから、動作モデルMの状態が競合を起こすことなく所定の入力状態から試験状態の集合Hに含まれる試験状態のうちの何れかへ遷移する過程に対応した試験命令列７０７を列挙する。７０６の部分は、動作モデルMの状態が試験命令列７０７に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、その次時刻状態を新たな入力状態として試験命令列列挙手段７０５に入力させる。

本発明の第１の実施例の全体構成図



【特許請求の範囲】

【請求項１】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを構成し、
該動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を列挙し、
前記動作モデルの状態がその初期状態から前記各試験状態へ遷移する過程に対応した試験命令列を生成する、
ことを特徴とするプロセッサの動作モデルと論理検証用試験命令列の自動生成方法。

【請求項２】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報を保持する仕様情報保持手段と、
該仕様情報保持手段が保持する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを構成する動作モデル構成手段と、
該動作モデル構成手段によって構成される動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を列挙する試験状態列挙手段と、
前記動作モデル構成手段によって構成される動作モデルの状態がその初期状態から前記試験状態列挙手段によって列挙される各試験状態へ遷移する過程に対応した試験命令列を生成する試験命令列生成手段と、
を有することを特徴とするプロセッサの動作モデルと論理検証用試験命令列の自動生成装置。

【請求項３】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを初期動作モデルとして構成し、
該初期動作モデルに含まれる各状態変数が有する複数の状態を、それらの状態が有する属性に基づいてグループ化することにより、前記各状態変数が有する状態の数を最小化し、該最小化が行われた動作モデルを構成する、
ことを特徴とするプロセッサの動作モデルの自動生成方法。

【請求項４】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報を保持する仕様情報保持手段と、
該仕様情報保持手段が保持する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを初期動作モデルとして構成する初期動作モデル構成手段と、
該初期動作モデル構成手段によって構成される初期動作モデルに含まれる各状態変数が有する複数の状態を、それらの状態が有する属性に基づいてグループ化することにより、前記各状態変数が有する状態の数を最小化し、該最小化が行われた動作モデルを構成する状態数最小化手段と、
を有することを特徴とするプロセッサの動作モデルの自動生成装置。

【請求項５】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数の状態が競合を起こす状態を含むことなく、前記動作モデルの状態が所定の入力状態から前記入力情報保持手段に保持される試験状態のうちの何れかへ遷移する過程に対応した試験命令列を列挙し、
前記入力情報として入力される動作モデルを構成する複数の状態変数の状態が、前記列挙された試験命令列に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、該次時刻状態を前記所定の入力状態として前記試験命令列を列挙する処理を繰り返し実行させる、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項６】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態が競合を起こすような状態である試験状態を、入力情報として保持する入力情報保持手段と、
該入力情報保持手段に保持された動作モデルを構成する複数の状態変数の状態が競合を起こす状態を含むことなく、前記動作モデルの状態が所定の入力状態から前記入力情報保持手段に保持される試験状態のうちの何れかへ遷移する過程に対応した試験命令列を列挙する試験命令列列挙手段と、
前記入力情報保持手段に保持された動作モデルを構成する複数の状態変数の状態が、前記試験命令列列挙手段が列挙した試験命令列に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、該次時刻状態を前記所定の入力状態として前記試験命令列列挙手段に入力させる次時刻状態計算手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項７】　プロセッサのパイプラインの構成及び前

記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態であって、前記状態評価関数の値が所定の条件を満たす状態を列挙し、
該列挙される状態に至る命令列を試験命令列として列挙する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項８】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として保持する入力情報保持手段と、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態であって、前記状態評価関数の値が所定の条件を満たす状態を列挙する到達可能状態列挙手段と、
該到達可能状態列挙手段によって列挙される状態に至る命令列を試験命令列として列挙する試験命令列列挙手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項９】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙し、
該列挙される状態に対応する前記状態評価関数の値を計算し、
前記列挙される状態に至る命令列を列挙し、
前記計算される値が所定の条件を満たす状態を前記列挙される状態から選択し、
該選択された状態に対応する命令列を前記列挙される命令列から選択し、
該選択された命令列を試験命令列として選択する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項１０】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態を入力とする状態評価関数を、入力情報として保持する入力情報保持手段と、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する到達可能状態列挙手段と、
該到達可能状態列挙手段によって列挙される状態に対応する前記状態評価関数の値を計算する状態評価関数計算手段と、
前記到達可能状態列挙手段によって列挙される状態に至る命令列を列挙する命令列列挙手段と、
前記状態評価関数計算手段によって計算される値が所定の条件を満たす状態を前記到達可能状態列挙手段によって列挙される状態から選択し、該選択された状態に対応する命令列を前記命令列列挙手段から選択し、該選択された命令列を試験命令列として出力する試験命令列出力手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項１１】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として入力し、
前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記入力情報として入力された状態の時系列を列挙し、
該列挙される状態の時系列を実現する命令列を試験命令列として列挙する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項１２】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として保持する入力情報保持手段と、
前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記入力情報として入力される状態

の時系列を列挙する到達可能状態時系列列挙手段と、
該到達可能状態時系列列挙手段によって列挙される状態の時系列を実現する命令列を試験命令列として列挙する試験命令列列挙手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項１３】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙し、
前記列挙される状態に至る命令列を列挙し、
前記列挙された状態からバックトラックしながら、前記入力情報として入力される状態の時系列に一致する状態の時系列を検索し、
該検索された状態の時系列の最終状態に至る命令列を前記列挙された命令列から選択し、
該選択された命令列を試験命令列として出力する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項１４】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列を、入力情報として保持する入力情報保持手段と、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する到達可能状態列挙手段と、
前記到達可能状態列挙手段によって列挙される状態に至る命令列を列挙する命令列列挙手段と、
前記到達可能状態列挙手段によって列挙された状態からバックトラックしながら、前記入力情報として入力される状態の時系列に一致する状態の時系列を検索する状態時系列検索手段と、
該状態時系列検索手段によって検索された状態の時系列の最終状態に至る命令列を前記命令列列挙手段から選択し、該選択された命令列を試験命令列として出力する試験命令列出力手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項１５】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として入力し、
前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記評価関数の値が所定の条件を満たす状態の時系列を列挙し、
該列挙される状態の時系列を実現する命令列を試験命令列として列挙する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項１６】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として保持する入力情報保持手段と、
前記入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、前記評価関数の値が所定の条件を満たす状態の時系列を列挙する到達可能状態時系列列挙手段と、
該到達可能状態時系列列挙手段によって列挙される状態の時系列を実現する命令列を試験命令列として列挙する試験命令列列挙手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【請求項１７】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として入力し、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙し、
前記列挙される状態に至る命令列を列挙し、
前記列挙された状態からバックトラックして得られる状態の時系列について、前記評価関数を計算し、
該計算される値が所定の条件を満たす状態の時系列を前記列挙された状態からバックトラックして得られる状態の時系列のうちから選択し、
該選択された状態の時系列の最終状態に至る命令列を前記列挙された命令列から選択し、
該選択された命令列を試験命令列として出力する、
ことを特徴とする論理検証用試験命令列の自動生成方法。

【請求項１８】　プロセッサのパイプラインの構成及び前記プロセッサが実行する各命令に関する仕様情報に基づき、前記パイプラインの動作を表現し、前記パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数により表現される動作モデルと、該動作モデルに含まれる複数の状態変数の状態の時系列の評価関数を、入力情報として保持する入力情報保持手段と、
該入力情報として入力される動作モデルを構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する到達可能状態列挙手段と、
前記到達可能状態列挙手段によって列挙される状態に至る命令列を列挙する命令列列挙手段と、
前記到達可能状態列挙手段によって列挙された状態からバックトラックして得られる状態の時系列について、前記評価関数を計算する状態時系列評価関数計算手段と、該状態時系列評価関数計算手段によって計算される値が所定の条件を満たす状態の時系列を前記到達可能状態列挙手段によって列挙された状態からバックトラックして得られる状態の時系列のうちから選択し、該選択された状態の時系列の最終状態に至る命令列を前記命令列列挙手段から選択し、該選択された命令列を試験命令列として出力する試験命令列出力手段と、
を有することを特徴とする論理検証用試験命令列の自動生成装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、パイプライン処理機能を有するプロセッサの動作論理を検証するための命令列を自動生成する技術に関する。

【０００２】

【従来の技術】プロセッサの機能又は構造の設計が仕様を正しく実現しているか否かを検証するための論理シミュレーション、及び製造されたプロセッサが仕様通りに動作するか否かを検証するための動作試験を行うためには、プロセッサに与えられる試験命令列（テストプログラム）の作成が不可欠である。更に、短時間で信頼性の高い検証を行うためには、質の良いテストプログラムを作成する必要がある。

【０００３】ここで、プロセッサの内部で命令のパイプライン処理を行うことは、処理速度の高いプロセッサを実現するために欠かせない技法である。パイプライン処理が実行される場合、同時に処理される複数の命令の間で、次のような競合状態（パイプラインハザード）が発生する可能性がある。

【０００４】・ある機能を有するハードウエア資源の数がＮである場合に、合計でＮ＋１以上の当該ハードウエア資源の使用要求が発生する状態。

・２つの命令間のレジスタ又はメモリに対するデータの読出し及び書込みの順序が入れ替わる結果、正常な実行結果が得られなくなる状態。

これらのパイプラインハザードを解決するためには、プロセッサ内に、パイプラインのインターロック及び命令の実行順序の入換え等を制御する機構が必要である。

【０００５】このような機構を実現するためのプロセッサの機能又は構造の設計が仕様を正しく実現しているか否かを検証するためには、論理シミュレーションを実行する必要がある。更に、製造されたプロセッサが仕様通りに動作するか否かを検証するために、動作試験を行う必要がある。そして、これらの検証を行うためには、プロセッサに与える試験命令列（テストプログラム）の作成が不可欠となる。

【０００６】プロセッサの構造は、一般に、実際のデータ処理を扱う演算部（ＡＬＵ、レジスタファイル等）と、それらを制御する制御部に分けることができる。そして、そのような構造のうちでも、制御部が有する機能は複雑である。そのため、その機能を効率よく検証できるテストプログラムを作成することは、重要であると同時に難しい問題である。

【０００７】従来のテストプログラムは、次のような方法の組み合わせによって作成されている。

方法１：試験命令列を人手で作成する。

【０００８】方法２：計算機を利用して、無作為に多くの試験命令列を発生させる。

方法３：既存のプログラムを利用して試験命令列を作成する。

なお、プロセッサの仕様を表現する記述からプロセッサの制御機構を検証するための試験命令列を自動生成する方式に関しては、本願出願人により既に２件の特許出願：特願平４－１４５８１２、特願平５－２１７８１３がなされているが、これらは本発明が実現しているような厳密な動作モデルに基づいたものではない。

【０００９】

【発明が解決しようとする課題】しかし、上述した方法１に基づく従来技術は、人手を利用するため、多くの開発期間を必要とする。更に、人手で作成された試験命令列は、その質を評価することが難しく、その有効性を客観的に保証することができないという問題点を有している。

【００１０】また、上述した方法２又は３に基づく従来技術では、命令数を増やすことによって検証の信頼性を上げることが可能ではある。しかし、論理シミュレーションや動作試験を行うためには、膨大な実行時間が必要となるという問題点を有している。

【００１１】更に、本願出願人により既に特許出願されている方式は、パイプラインハザードの発生時に前段側のパイプラインステージが停止するような制御を検証するための試験命令列の生成方式に限定されるものである。従って、これらの方式は、現在多く出現しつつある複雑なパイプライン制御方式に対応しきれないという問題点を有している。

【００１２】本発明は、より多くの種類のプロセッサを対象として、対象となるプロセッサのための論理シミュレーション及び動作試験の実行時間の短縮と、それらの検証の信頼性の向上を実現するために、プロセッサの動作論理を検証するための質の高い試験命令列を自動的かつ短時間に生成可能とすることを目的とする。

【００１３】

【課題を解決するための手段】図１は、本発明の第１の態様のブロック図である。まず、仕様情報保持手段１０２は、プロセッサのパイプラインの構成及びプロセッサが実行する各命令に関する仕様情報１０１を保持する。ここで、“パイプライン”とは、命令の処理を基本的に１クロックサイクル（命令サイクル）で処理できる幾つかの単位動作（パイプラインステージ）に分割し、連続する複数の入力に対する処理実行をオーバーラップさせて各パイプラインステージにおいて並列に処理する機能をいう。１つの命令を１命令サイクルの間保持することのできるハードウエア単位をパイプラインユニットとよぶ。また、“プロセッサが実行する各命令についての仕様情報”とは、プロセッサのパイプライン動作を決定する情報であって、例えば、各命令のパイプライン処理の経路や、各命令のパイプラインステージ毎のハードウエア資源の占有についての情報、及びレジスタ又はメモリ等に対するデータの読出し／書込みの実行についての情報である。

【００１４】動作モデル構成手段１０４は、仕様情報保持手段１０１が保持する仕様情報１０１に基づき、パイプラインの動作を表現し、パイプラインの各構成部分（各パイプラインユニット）の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデル１０３を構成する。この動作モデル１０３において、各パイプラインユニットに入力する命令は、外部又は他の状態変数から１つの状態変数に入力される状態に対応付けられる。

【００１５】試験状態列挙手段１０６は、動作モデル構成手段１０４によって構成される動作モデル１０３に含まれる複数の状態変数の状態が競合を起こすような状態である試験状態１０５を列挙する。

【００１６】試験命令列生成手段１０８は、動作モデル構成手段１０４によって構成される動作モデル１０３がその初期状態から試験状態列挙手段１０６によって列挙される各試験状態１０５へ遷移する過程に対応した試験命令列１０７を生成する。

【００１７】上述の本発明の第１の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図２は、本発明の第２の態様のブロック図である。

【００１８】仕様情報保持手段２０２は、プロセッサのパイプラインの構成及びプロセッサが実行する各命令に関する仕様情報２０１を保持する。初期動作モデル構成手段２０４は、仕様情報保持手段２０２が保持する仕様情報２０１に基づいて、パイプラインの動作を表現し、パイプラインの各構成部分の状態とそれらの状態の遷移関係を複数の状態変数によって表現する動作モデルを初期動作モデル２０３として構成する。

【００１９】状態数最小化手段２０６は、初期動作モデル構成手段２０４によって構成される初期動作モデル２０３に含まれる各状態変数が有する複数の状態を、それらの状態が有する属性に基づいてグループ化することにより、各状態変数が有する状態の数を最小化し、その最小化が行われた動作モデル２０５を構成する。

【００２０】上述の本発明の第２の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図３は、本発明の第３の態様のブロック図である。

【００２１】入力情報保持手段３０３は、上述した本発明の第１又は第２の態様によって構成される動作モデル３０１（動作モデル１０３、２０５に対応する）と、その動作モデル３０１に含まれる複数の状態変数の状態が競合を起こすような状態である試験状態３０２を、入力情報として保持する。

【００２２】試験命令列列挙手段３０６は、入力情報保持手段３０３に保持された動作モデル３０１を構成する複数の状態変数の状態が競合を起こす状態を含むことなく、動作モデル３０１の状態が所定の入力状態３０４から入力情報保持手段３０３に保持される試験状態３０２のうちの何れかへ遷移する過程に対応した試験命令列３０５を列挙する。

【００２３】次時刻状態計算手段３０８は、入力情報保持手段３０２に保持された動作モデル３０１を構成する複数の状態変数の状態が、試験命令列列挙手段３０６が列挙した試験命令列３０５に対応する試験状態になった後に次に遷移する状態である次時刻状態３０７を計算し、その次時刻状態３０７を所定の入力状態３０４として試験命令列列挙手段３０６に入力させる。

【００２４】上述の本発明の第３の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図４(a)は、本発明の第４の態様のブロック図である。

【００２５】入力情報保持手段４０３は、上述した本発明の第１又は第２の態様によって構成される動作モデル４０１（動作モデル１０３、２０５に対応する）と、動作モデル４０１に含まれる複数の状態変数の状態を入力とする状態評価関数４０２を、入力情報として保持する。

【００２６】到達可能状態列挙手段４０４は、入力情報として入力される動作モデル４０１を構成する複数の状態変数についてその初期状態から到達可能な状態であって、状態評価関数４０２の値が所定の条件を満たす状態を列挙する。

【００２７】試験命令列列挙手段４０６は、到達可能状態列挙手段４０４によって列挙される状態に至る命令列を試験命令列４０５として列挙する。図４(b)は、本発明の第４の態様のより具体的なブロック図である。

【００２８】入力情報保持手段４０９は、上述した本発明の第１又は第２の態様により構成される動作モデル４０７（動作モデル１０３、２０５に対応する）と、動作モデル４０７に含まれる複数の状態変数の状態を入力とする状態評価関数４０８を、入力情報として保持する。

【００２９】到達可能状態列挙手段４１０は、入力情報として入力される動作モデル４０７を構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する。

【００３０】状態評価関数計算手段４１１は、到達可能状態列挙手段４１０によって列挙される状態に対応する状態評価関数４０８の値を計算する。命令列列挙手段４１２は、到達可能状態列挙手段４１０によって列挙される状態に至る命令列を列挙する。

【００３１】試験命令列出力手段４１４は、状態評価関数計算手段４１１によって計算される値が所定の条件を満たす状態を到達可能状態列挙手段４１０によって列挙される状態から選択し、選択された状態に対応する命令列を命令列列挙手段４１２から選択し、選択された命令列を試験命令列４１３として出力する。

【００３２】上述の本発明の第４の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図５(a)は、本発明の第５の態様のブロック図である。

【００３３】入力情報保持手段５０３は、上述した本発明の第１又は第２の態様によって構成される動作モデル５０１（動作モデル１０３、２０５に対応する）と、動作モデル５０１に含まれる複数の状態変数の状態の時系列５０２を、入力情報として保持する。

【００３４】到達可能状態時系列列挙手段５０４は、入力情報として入力される動作モデル５０１を構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、入力情報として入力される状態の時系列５０２を列挙する。

【００３５】試験命令列列挙手段５０６は、到達可能状態時系列列挙手段５０４によって列挙される状態の時系列を実現する命令列を試験命令列５０５として列挙する。図５(b)は、本発明の第５の態様のより具体的なブロック図である。

【００３６】入力情報保持手段５０９は、上述した本発明の第１又は第２の態様により構成される動作モデル５０７（動作モデル１０３、２０５に対応する）と、動作モデル５０７に含まれる複数の状態変数の状態の時系列５０８を、入力情報として保持する。

【００３７】到達可能状態列挙手段５１０は、入力情報として入力される動作モデル５０７を構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する。

【００３８】命令列列挙手段５１１は、到達可能状態列挙手段５１０によって列挙される状態に至る命令列を列挙する。状態時系列検索手段５１２は、到達可能状態列挙手段５１０によって列挙された状態からバックトラックしながら、入力情報として入力される状態の時系列５０８に一致する状態の時系列を検索する。

【００３９】試験命令列出力手段５１４は、状態時系列検索手段５１２によって検索された状態の時系列の最終状態に至る命令列を命令列列挙手段５１１から選択し、選択された命令列を試験命令列５１３として出力する。

【００４０】上述の本発明の第５の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。図６(a)は、本発明の第６の態様のブロック図である。

【００４１】入力情報保持手段６０３は、上述した本発明の第１又は第２の態様によって構成される動作モデル６０１（動作モデル１０３、２０５に対応する）と、動作モデル６０１に含まれる複数の状態変数の状態の時系列の評価関数６０２を、入力情報として保持する。

【００４２】到達可能状態時系列列挙手段６０４は、入力情報として入力される動作モデル６０１を構成する複数の状態変数についてその初期状態から到達可能な状態の時系列であって、評価関数６０２の値が所定の条件を満たす状態の時系列を列挙する。

【００４３】試験命令列列挙手段６０６は、到達可能状態時系列列挙手段６０４によって列挙される状態の時系列を実現する命令列を試験命令列６０５として列挙する。図６(b)は、本発明の第６の態様のより具体的なブロック図である。

【００４４】入力情報保持手段６０９は、上述した本発明の第１又は第２の態様により構成される動作モデル６０７（動作モデル１０３、２０５に対応する）と、動作モデル６０７に含まれる複数の状態変数の状態の時系列の評価関数６０８を、入力情報として保持する。

【００４５】到達可能状態列挙手段６１０は、入力情報として入力される動作モデル６０７を構成する複数の状態変数についてその初期状態から到達可能な状態を列挙する。

【００４６】命令列列挙手段６１１は、到達可能状態列挙手段６１０によって列挙される状態に至る命令列を列挙する。状態時系列評価関数計算手段６１２は、到達可能状態列挙手段６１０によって列挙された状態からバックトラックして得られる状態の時系列について、その評価関数６０８を計算する。

【００４７】試験命令列出力手段６１４は、状態時系列評価関数計算手段６１２によって計算される値が所定の条件を満たす状態の時系列を到達可能状態列挙手段６１０によって列挙された状態からバックトラックして得ら

れる状態の時系列のうちから選択し、選択された状態の時系列の最終状態に至る命令列を命令列列挙手段６１１から選択し、選択された命令列を試験命令列６１３として出力する。

【００４８】上述の本発明の第６の態様として示される装置が実現する機能は、その機能を実現する方法として具体化することもできる。

【００４９】

【作用】本発明の第１の態様では、入力された仕様情報１０１に基づきプロセッサのパイプラインの動作を表現する動作モデル１０３が自動的に構成される。この動作モデル１０３が正確かつ簡略に構成されることにより、出力される試験命令列１０７の品質向上と、その命令列に基づくプロセッサ試験の処理の効率化を実現することができる。

【００５０】次に、試験状態１０５が仕様情報１０１に基づき列挙された後、初期状態から試験状態１０５への遷移過程に対応した試験命令列１０７が生成される。試験状態１０５の選び方により、目的に応じたテストプログラムを自由に作ることが可能となる。

【００５１】本発明の第２の態様では、入力された仕様情報２０１に直接的に対応した動作モデルが初期動作モデル２０３として生成された後、その動作モデルに含まれる各状態変数が有する複数の状態がグループ化され、複数の状態が１つの状態として処理されることにより、動作モデルの状態数が削減される。この結果、パイプラインの構成、及びパイプライン上の命令の流れに関する仕様を維持したまま、プロセッサの簡約化された動作モデル２０５を得ることができる。そして、このような動作モデル２０５に基づいて試験命令列を生成するための処理に擁する時間を短縮することができる。

【００５２】本発明の第３の態様では、入力された動作モデル３０１及びその動作モデル３０１の試験状態３０２に基づいて、試験状態３０２を含まない状態遷移により試験状態３０２に到達する試験命令列３０５を列挙する処理と、試験状態３０２の次時刻（次の命令サイクル）の状態を求める処理が繰り返し実行されることにより、試験命令列３０５が自動的に生成される。次時刻に複雑な状態遷移がある場合が試験状態３０２として設定されることにより、試験状態３０２を含まない状態遷移は単純なものとなって、試験命令列３０２を効率良く抽出することができる。

【００５３】本発明の第４の態様では、動作モデル４０１、４０７に対して試験項目を指定するための入力情報として、或る１つの時刻における試験状態の集合ではなく、状態評価関数４０２、４０８を指定できる。

【００５４】パイプラインプロセッサでは、トラップ制御において、トラップ発生時にトラップを誘発させた命令が実行されるより前に実行された命令と、それより後に実行された命令を正確に区別して制御する機構が、重要な役割を果たす。特に、厳密トラップ（precisetrap）の実現のためには、トラップを誘発させた命令が実行されるより前に実行された命令は、トラップハンドラに制御が移った時点において実行を完了していなければならず、かつ、トラップを誘発させた命令が実行されるより後に実行されるはずの命令は、トラップハンドラに制御が移った時点において実行が開始されていてはならない。このような状態を検証するためには、例えば、パイプラインにできる限り多くの命令が詰った状態かつトラップを発生させる状態が試験項目となるように、試験命令列を生成する必要がある。

【００５５】本発明の第４の態様では、このような定性的に表現される試験項目を、状態を評価する状態評価関数４０２、４０８によって効率的に定義することができ、それに対応する試験命令列５０５、５１３を、自動的かつ効率的に生成することができる。

【００５６】本発明の第５の態様では、動作モデル５０１、５０７に対して試験項目を指定するための入力情報として、状態の時系列５０２、５０８を指定できる。例えば或る種のパイプラインプロセッサにおいて、命令の実行結果の書込みユニットとレジスタファイルとの間にバッファが設けられ、このバッファによって実際のライトバック動作がプログラム上のステップの順番に対応するように修正されることにより、正確なトラップが実現されるように設計される場合がある。このようなパイプライン機構を検証するためには、数命令サイクルの間命令の完了順序が入れ替わる状態を発生させ、バッファにデータを保持させた後に、トラップを発生させるような試験命令列を生成する必要がある。

【００５７】本発明の第５の態様では、このような状態の変化を、状態の時系列５０２、５０８として指定することができ、一定期間にわたり特定の状態を発生させるような試験命令列５０５、５１３を、自動的かつ効率的に生成することができる。

【００５８】本発明の第６の態様では、動作モデル６０１、６０７に対して試験項目を指定するための入力情報として、状態の時系列の評価関数６０２、６０８を指定できる。

【００５９】例えば、プログラム上のレジスタ番号と物理レジスタ番号との間で動的に番号の付替えを行う、いわゆるレジスタリネーミング機構を有するパイプラインプロセッサが存在するが、このようなプロセッサにおいては、リネーミングのための物理レジスタの数（リネーミングスペース）が不足する場合が問題となる。この不足が生じないように設計されたプロセッサにおいて本当にこの不足が発生しないことを検証するためには、できるだけ多くのリネーミングスペースが消費されるような試験命令列が必要となる。

【００６０】本発明の第６の態様では、このような一定期間の状態が定性的に表現される試験項目を、状態の時

系列を評価する評価関数６０２、６０８によって効率的に定義することができ、それに対応する試験命令列６０５、６１３を、自動的かつ効率的に生成することができる。

【００６１】

【実施例】以下、図面を参照しながら本発明の実施例につき詳細に説明する。

＜第１の実施例＞図７は、本発明の第１の実施例の全体構成図である。この構成図によって実現される機能は、特には図示しないプロセッサとメモリ、及びそれらによって実現される制御プログラムを主な構成要素とするコンピュータシステムによって実現されている。

【００６２】図７において、仕様情報保持部７０１は、検証の対象であるプロセッサのパイプライン構成及びプロセッサが実行するそれぞれの命令に関する仕様情報を保持する。

【００６３】初期動作モデル構成部７０２は、仕様情報保持部７０１に保持された仕様情報から、パイプラインに関する初期動作モデルM_0を構成する。状態数最小化部７０３は、初期動作モデルM_0の状態の数を最小化することにより、動作モデルMを構成する。

【００６４】入力情報保持部７０４は、上述の動作モデルMとそのモデルに対する試験状態の集合Hとからなる入力情報を保持する。試験命令列列挙部７０５は、入力情報保持部７０４に保持された動作モデルMと試験状態の集合Hとから、入力情報保持部７０４に保持された動作モデルMを構成する複数の状態変数の状態が競合を起こす状態を含むことなく、動作モデルMの状態が所定の入力状態から入力情報保持部７０４に保持される試験状態の集合Hに含まれる試験状態のうちの何れかへ遷移する過程に対応した試験命令列７０７を列挙する。

【００６５】次時刻状態計算部７０６は、入力情報保持部７０４に保持された動作モデルMを構成する複数の状態変数の状態が、試験命令列列挙部７０５が列挙した試験命令列７０７に対応する試験状態になった後に次に遷移する状態である次時刻状態を計算し、その次時刻状態を新たな入力状態として試験命令列列挙手段７０５に入力させる。

【００６６】以上の構成を有する第１の実施例の動作について、図８～図１５の動作フローチャート、図１６及び図１７のデータ構成図、並びに、図１８～図２８の動作説明図に従って、詳細に説明する。

【００６７】まず、図８のステップ８０１で示されるように、図７に示される仕様情報保持部７０１に、仕様情報が入力される。図１８に、仕様情報によって表されるパイプライン構成の例を示す。

【００６８】パイプライン処理では、１つの命令によって実行される１つの処理過程が複数の時間的に独立した処理ステージに分割され、それぞれの命令のそれぞれの処理ステージはそれぞれに対応した処理ユニットによって実行される。そして、それぞれの命令サイクルにおいて、複数の処理ユニットによって、複数の命令に対応する複数の相互に異なる処理ステージが並列に実行される。

【００６９】図１８において、命令フェッチユニット（ＩＦユニット）１８０１は、それぞれの命令の命令フェッチステージ（ＩＦステージ）を実行するユニットである。このユニットは、特には図示しないメモリから命令をフェッチする。

【００７０】命令デコードユニット（ＩＤユニット）１８０２は、それぞれの命令の命令デコードステージ（ＩＤステージ）を実行する。このユニットは、フェッチされた命令をデコードし、また、その命令に対応するオペランドをレジスタから読み出す。

【００７１】算術論理演算ユニット（ＡＬＵユニット）１８０３は、それぞれの命令の算術論理演算ステージ（ＡＬＵステージ）を実行する。このユニットは、デコードされた命令が整数演算命令又は論理演算命令である場合に、その命令に対応する整数演算又は論理演算を実行する。

【００７２】メモリアクセスユニット（ＭＥＭユニット）１８０４は、それぞれの命令のメモリアクセスステージ（ＭＥＭステージ）を実行する。このユニットは、ＡＬＵユニット１８０３によりアドレスが計算されたデータを、特には図示しないメモリから読み出し又はそのメモリに書き込む。

【００７３】F_1ユニット１８０５、F_2ユニット１８０６、及びF_3ユニット１８０７とから構成される浮動小数点演算ユニット（ＦＰＵユニット）は、それぞれの命令の浮動小数点演算ステージ（ＦＰＵステージ）を実行する。このユニットは、デコードされた命令が浮動小数点演算命令である場合に、その命令に対応するオペランドに対して浮動小数点演算を実行する。

【００７４】ライトバックユニット（ＷＢユニット）１８０８は、それぞれの命令のライトバックステージ（ＷＢステージ）を実行する。このユニットは、ＭＥＭユニット１８０４によって特には図示しないメモリから取得されたデータ又はＡＬＵユニット１８０３によって演算されたデータを、特には図示しないレジスタに書き込む。

【００７５】上述のパイプラインの構成において、１命令サイクルの期間にフェッチされる命令は１個であり、ＦＰＵステージ以外のステージの実行は、１命令サイクルで終了する。ＦＰＵステージが消費する命令サイクル数は実行される命令によって異なる。１つの命令がF_1ユニット１８０５によって実行された後にF_3ユニット１８０７によって実行される場合には、その命令に対応するＦＰＵステージは２命令サイクルを消費する。また、１つの命令がF_1ユニット１８０５によって実行された後にF_2ユニット１８０６が実行され更にその後に

F_3ユニット１８０７によって実行される場合には、その命令に対応するＦＰＵステージは３命令サイクルを消費する。

【００７６】上述した例で示されるようなパイプライン構成に関する仕様を所定の形式に従って記述した仕様情報が、仕様情報保持部７０１に入力される。次に、図１９に、仕様情報によって表される命令の実行形態の例を示す。この仕様情報により示される仕様は以下のとうりである。即ち、命令Ａは、命令サイクルの進行に同期して、その命令に対応するＩＦステージ、ＩＤステージ、ＡＬＵステージ、及びＷＢステージが順に実行される。また、命令Ｂは、命令サイクルの進行に同期して、その命令に対応するＩＦステージ、ＩＤステージ、ＡＬＵステージ、ＭＥＭステージ、及びＷＢステージが順に実行される。更に、命令Ｃは、命令サイクルの進行に同期して、その命令に対応するＩＦステージ、ＩＤステージ、F_1ユニット１８０５の実行タイミングとF_3ユニット１８０７の実行タイミングとからなるＦＰＵステージ、及びＷＢステージが順に実行される。加えて、命令Ｄは、命令サイクルの進行に同期して、その命令に対応するＩＦステージ、ＩＤステージ、F_1ユニット１８０５の実行タイミングとF_2ユニット１８０６の実行タイミングとF_3ユニット１８０７の実行タイミングとからなるＦＰＵステージ、及びＷＢステージが順に実行される。

【００７７】上述した例で示されるような命令の実行形態に関する仕様を所定の形式に従って記述した仕様情報が、仕様情報保持部７０１に入力される。次に、図８のステップ８０２で示されるように、図７に示される初期動作モデル構成部７０２は、仕様情報保持部７０１に保持された仕様情報に従って、初期動作モデルM_0を構成する。図１８及び図１９に示される仕様情報に基づいて、図２０に概念的に示される初期動作モデルM_0が構成される。

【００７８】仕様情報から初期動作モデルM_0を構成するためのアルゴリズムは、以下に示される。まず、パイプラインの構成例に関する仕様情報（図１８参照）に基づいて、パイプラインを構成する各ユニットに対応して、図１６に示される構造体として定義される状態変数が確保される。このとき、ユニット内部のパイプラインも考慮され、命令サイクル毎に命令を１個保持できるユニットにつき１個の状態変数が対応させられる。図１８のパイプラインの構成例では、ＩＦユニット１８０１、ＩＤユニット１８０２、ＡＬＵユニット１８０３、ＭＥＭユニット１８０４、及びＷＢユニット１８０８のそれぞれに対応して１つずつの状態変数が確保されると共に、ＦＰＵユニットを構成するF_1ユニット１８０５、F_2ユニット１８０６、及びF_3ユニット１８０７のそれぞれに対応して１つずつの状態変数が確保される。また、図２０に示される初期動作モデルM_0の例では、番号#1～#8で参照される円が状態変数を表しており、それぞれの円は、図１８のパイプライン構成の例において番号１８０１～１８０８で参照される各ユニットに、１対１に対応している。

【００７９】次に、上述のようにして確保された状態変数を構成する図１６に示される各構成要素が決定される。図１６において、状態変数の“前段に接続する状態変数のリスト”には、次時刻にその状態変数に対応するユニットが処理する状態を保持する可能性のあるユニットに対応する状態変数のリストが保持される。例えば図１８及び図１９の例において、ＡＬＵユニット１８０３に注目した場合に、そのユニットによって実行されるのは命令Ａ及び命令Ｂという２つの状態であって、ＡＬＵユニット１８０３がこれらの状態を実行する現在時刻より１単位前の時刻には、ＩＤユニット１８０２がこれらの状態を実行する。従って、ＡＬＵユニット１８０３に対応する状態変数の“前段に接続する状態変数のリスト”には、ＩＤユニット１８０２に対応する状態変数が登録される。また、F_3ユニット１８０７に注目した場合に、そのユニットによって実行されるのは命令Ｃ及び命令Ｄという２つの状態であって、F_3ユニット１８０７がこれらの状態を実行する現在時刻より１単位前の時刻には、F_1ユニット１８０５が命令Ｃを実行しF_2ユニット１８０６が命令Ｄを実行する。従って、F_3ユニット１８０７に対応する状態変数の“前段に接続する状態変数のリスト”には、F_1ユニット１８０５に対応する状態変数とF_2ユニット１８０６に対応する状態変数が登録される。

【００８０】図１６において、状態変数の“後段に接続する状態変数のリスト”には、その状態変数に対応するユニットが保持する状態を次時刻に処理する可能性のあるユニットに対応する状態変数のリストが保持される。例えば図１８及び図１９の例において、ＡＬＵユニット１８０３に注目した場合に、そのユニットにより実行されるのは命令Ａ及び命令Ｂという２つの状態であって、次時刻にこれらの状態を実行する可能性のあるユニットは、ＷＢユニット１８０８とＭＥＭユニット１８０４である。従って、ＡＬＵユニット１８０３に対応する状態変数の“後段に接続する状態変数のリスト”には、ＷＢユニット１８０８に対応する状態変数とＭＥＭユニット１８０４に対応する状態変数が登録される。

【００８１】それぞれの状態変数の上述した“前段に接続する状態変数のリスト”と“後段に接続する状態変数のリスト”は、図２０に示される初期動作モデルM_0の例では、それぞれの円で示されるそれぞれの状態変数の間を結ぶ矢印を規定する。

【００８２】図１６において、状態変数の“入力の集合”には、その状態変数に対応するユニットにパイプラインの外部から入力される状態の集合が保持される。初期動作モデルM_0においては、命令フェッチユニットに

対応する状態変数の“入力の集合”は、そのパイプラインにおいて実行可能な全ての命令の集合であり、その他のユニットに対応する状態変数の“入力の集合”は、空集合である。より具体的には、例えば図１８及び図１９の例において、ＩＦユニット１８０１に対応する状態変数の“入力の集合”には、命令Ａ、Ｂ、Ｃ、及びＤという４つの状態が登録される。また、図２０に示される初期動作モデルM_0の例では、ＩＦユニット１８０１に対応する状態変数の“入力の集合”によって、#1で参照される円で示されるＩＦユニット１８０１に対応する状態変数に入力される矢印と、その矢印に付加される状態（命令）が規定される。

【００８３】図１６において、状態変数の“状態の集合”には、その状態変数に対応するユニットが処理する状態の集合が保持される。なお、ここには、処理する状態（命令）が無いことを示す情報も保持される。初期動作モデルM_0においては、各状態変数の“状態の集合”は、各状態変数に対応するユニットに入力する可能性のある全ての命令と命令が無いことを示す情報とからなる。例えば図１８及び図１９の例において、ＡＬＵユニット１８０３に注目した場合に、そのユニットにより実行されるのは命令Ａ及び命令Ｂという２つの状態である。従って、ＡＬＵユニット１８０３に対応する状態変数の“状態の集合”には、命令Ａ、命令Ｂという２つの状態と、命令が無いことを示す情報が登録される。また、図２０に示される初期動作モデルM_0の例では、上述した“状態の集合”は、それぞれの状態変数に対応する円の中の記号として表現される。ここで、記号Ａ～Ｄは命令Ａ～Ｄという４つの状態に対応し、記号Ｏは処理する命令が無いことを示す情報に対応する。なお、後述する状態数最小化処理によって得られる動作モデルＭにおいては、各状態変数の“状態の集合”には、グループ化によって得られる状態の集合が保持され得る。

【００８４】図１６において、状態変数の“遷移関数”は、その状態変数に対応するユニットの前段に接続されるユニットから出力され又はパイプラインの外部から入力される状態からその状態変数に対応するユニットが次時刻に処理する状態を得るための関数が保持される。例えば図１８及び図１９の例において、ＷＢユニット１８０８に対応する状態変数の“遷移関数”によって、ＷＢユニット１８０８の前段に接続されるＡＬＵユニット１８０３が命令Ａを出力した場合にその命令Ａを次時刻に処理する命令として取得すること、ＷＢユニット１８０８の前段に接続されるＭＥＭユニット１８０４が命令Ｂを出力した場合にその命令Ｂを次時刻に処理する命令として取得すること、及びＷＢユニット１８０８の前段に接続されるF_3ユニット１８０７が命令Ｃを出力した場合にその命令Ｃを次時刻に処理する命令として取得することが規定される。また、図２０に示される初期動作モデルM_0の例では、上記“遷移関数”は、１つの状態変数から他の１つの状態変数へ向かう、記号Ａ、Ｂ、Ｃ、又はＤの組合せによってラベル付けされた矢印によって表現されている。

【００８５】図１６において、状態変数の“動作テーブル”には、図１７に示されるデータ構造を有する動作テーブルが保持される。この動作テーブルは、その状態変数に対応するユニットが状態$_1$、状態$_2$、状態$_3$、・・・等として示されるそれぞれの状態にある場合（それぞれの命令を実行している場合）に、その状態変数に対応するユニットが動作$_1$、動作$_2$、動作$_3$、・・・等として示されるレジスタの読出し又は書込み等のパイプラインの制御に影響する動作を実行するか否か、又は当該ユニットが特定のハードウエア資源を占有するか否か等を定義する。例えば図１８及び図１９の例において、ＭＥＭユニット１８０４に対応する状態変数の“動作テーブル”には、全ての状態に対してメインメモリが占有される旨が設定される。この動作テーブルは、後述する状態数最小化の処理において使用される。

【００８６】それぞれの状態変数に対応して図１６に示されるデータ構造を構成する上述したそれぞれの集合又は“遷移関数”は、２分決定グラフ（ＢＤＤ）を用いて効率良く表現し処理することができる。なお、図１６に示されるデータ構造を構成するそれぞれの構成要素は、それぞれの構成要素に対応する機能が定義されるデータへのポインタであってもよい。

【００８７】続いて、図８のステップ８０３で示されるように、図７に示される状態数最小化部７０３は、上述のようにして構成された初期動作モデルM_0の状態の数を最小化することにより、動作モデルＭを構成する。

【００８８】図９に、ステップ８０３の状態数最小化処理に対応する更に詳細な動作フローチャートを示す。まず、図９のステップ９０１において、図８のステップ８０２で構成された初期動作モデルM_0の各状態変数が順序付けされる。具体的には、初期動作モデルM_0を構成する入力段の状態変数から最終段の状態変数に向かって、それぞれの状態変数に、その接続順に対応する順序情報が付加される。図２０に示される初期動作モデルM_0の例では、この順序付け処理により決定される状態変数の順序は、#1、#2、#3、#5、#4、#6、#7、#8の状態変数の順になる。

【００８９】次に、図９のステップ９０２で、入力段の状態変数から最終段の状態変数に向かって順に、それぞれの状態変数における状態数が最小化される処理が実行される。図１０に、この処理に対応する更に詳細な動作フローチャートを示す。

【００９０】まず、図１０のステップ１００１で、入力段の状態変数において、“状態の集合”に含まれる状態のうち“動作テーブル”における値が同じ状態が、グループ化される。即ち、入力段の状態変数に対応するパイプラインユニットにおいて、そのユニットで実行される

状態（命令）のうち、占有する資源、又はレジスタの読出し若しくは書込みの動作等が同じ状態がグループ化される。この結果、グループ化の対象となった状態が新たな１つの状態に置き換えられる。即ち、状態変数の“状態の集合”において、グループ化の対象となった状態が削除され、グループ化の結果を示す新たな１つの状態が登録される。また、状態変数の“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな１つの状態に関する記述が追加される。この記述は、グループ化の対象となった状態に関して共通に記述されていた内容と同じ内容を有する。

【００９１】図１８の例に示されるように、通常、入力段の状態変数は命令フェッチユニット（図１８ではＩＦユニット１８０１）に対応しており、このユニットによってフェッチされる全ての命令に対応する全ての状態の“動作テーブル”における値は同じである。このため、例えば図２１に示されるように、#1で示される入力段の状態変数において、全ての命令（全ての状態）が新たな１つの状態Ｘに置き換えられる。そして、#1の状態変数の“状態の集合”において、グループ化の対象となった状態Ａ、Ｂ、Ｃ、Ｄ（図２０参照）が削除され、グループ化の結果を示す新たな１つの状態Ｘが登録される。また、状態変数の“動作テーブル”において、そのグループを構成する状態Ａ、Ｂ、Ｃ、Ｄに関する記述が削除されて、グループ化の結果を示す新たな１つの状態Ｘに関する記述が追加される。この記述は、グループ化の対象となった状態Ａ、Ｂ、Ｃ、Ｄに関して記述されていた共通の内容と同じ内容を有する。なお、記号Ｏで示される、処理する状態（命令）が無いことを示す情報は、グループ化されずに残される。

【００９２】次に、図９のステップ９０１で決定された順序に従って、入力段の次段の状態変数から最終段の状態変数に向かって、ステップ１００２～ステップ１００５の処理が繰り返し実行される。

【００９３】ステップ１００２では、図９のステップ９０１で決定された順序に従って、入力段の状態変数の最も近くに接続され、かつ処理されていない状態変数Ｕが選択される。

【００９４】ステップ１００３では、状態変数Ｕの“状態の集合”に含まれる状態のうち、状態変数Ｕの“前段に接続する状態変数のリスト”に含まれる状態変数の“状態の集合”において同じグループに属し、かつ状態変数Ｕの“動作テーブル”における値が同じ状態が、グループ化される。この結果、ステップ１００１の場合と同様にして、グループ化の対象となった状態が新たな１つの状態に置き換えられる。即ち、状態変数Ｕの“状態の集合”において、グループ化の対象となった状態が削除されて、グループ化の結果を示す新たな１つの状態が登録される。また、状態変数Ｕの“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな１つの状態に関する記述が追加される。そして、この記述は、グループ化の対象となった状態に関して記述されていた共通の内容と同じ内容を有する。上述のグループ化の処理により、グループ化が行われた状態変数の前段に接続される状態変数に対するグループ化の結果と同じ結果、又はその結果を更に細分化した結果が得られる。

【００９５】ステップ１００４では、ステップ１００３によるグループ化の処理が試みられた状態変数の“状態の集合”に、その状態変数の前段に接続される状態変数の“状態の集合”に含まれる状態とは異なる状態が含まれる場合、即ち、ステップ１００３によるグループ化の処理が試みられた状態変数の“状態の集合”に、そのグループ化の処理が試みられた状態変数の前段に接続される状態変数の“状態の集合”に含まれる状態を細分化した状態が含まれている場合は、ステップ１００３によるグループ化の処理が試みられた状態変数に対して、パイプラインの外部から、上述の異なる状態の入力があると仮定され、その状態変数の“入力の集合”に、その外部入力に対応する状態が追加される。外部入力が追加される理由は、ステップ１００３によるグループ化の処理が試みられた状態変数に、その状態変数の前段に接続される状態変数の“状態の集合”に含まれる状態が入力された場合において、“遷移関数”が新たな状態を決定できるようにするためである。なお、ステップ１００４の処理は、ステップ１００３によるグループ化の処理の結果、グループ化が実際に行われた状態変数と、グループ化が実際には行われなかった状態変数の両方を対象として実行される。

【００９６】上述したステップ１００２～１００４の処理が、ステップ１００５で、処理されていない状態変数がなくなったと判定されるまで、即ち、最終段の状態変数が処理されるまで繰り返し実行される。

【００９７】図１０の動作フローチャートで示される図９のステップ９０２の入力段側からの状態数最小化の処理によって、例えば図２０に示される初期動作モデルM_0から図２１に示される動作モデルM_1が得られる。この動作モデルM_1において、前述したように、入力段の#1の状態変数において、状態Ａ、Ｂ、Ｃ、Ｄがグループ化されることによって、それらの状態が新たな状態Ｘに置き換えられる。また、#2の状態変数において、状態Ａ、Ｂがグループ化されることによってそれらの状態が新たな状態Ｙに置き換えられ、状態Ｃ、Ｄがグループ化されることによってそれらの状態が新たな状態Ｚに置き換えられる。この場合、#2の状態変数の“状態の集合”に新たに含まれた状態Ｙ、Ｚは、#2の状態変数の前段に接続される#1の状態変数の“状態の集合”に含まれる状態Ｘとは異なる。このため、外部入力として、#2の状態変数の“入力の集合”に、状態Ｙ、Ｚが追加される。更

に、#2の状態変数に対するグループ化の結果、#3の状態変数の“状態の集合”に含まれている状態Ａ、Ｂは、#3の状態変数の前段に接続される#2の状態変数の“状態の集合”に含まれる状態Ｙ、Ｚとは異なる結果となる。このために、外部入力として、#3の状態変数の“入力の集合”に、状態Ａ、Ｂが追加される。これと同様にして、#2の状態変数に対するグループ化の結果、#5の状態変数の“状態の集合”に含まれている状態Ｃ、Ｄも、#5の状態変数の前段に接続される#2の状態変数の“状態の集合”に含まれる状態Ｙ、Ｚとは異なる結果となる。このため、外部入力として、#5の状態変数の“入力の集合”に、状態Ｃ、Ｄが追加される。

【００９８】上述した図９のステップ９０２の入力段側からの状態数最小化の処理により、それぞれの状態変数に関して、入力段側からその状態変数に対応するユニットまでの動作が全て同じ状態（命令）をグループ化することができる。この結果、後述する試験命令列を生成するための処理が必要とする時間を短縮することができる。

【００９９】図９のステップ９０２の入力段側からの状態数最小化の処理の後、ステップ９０３で、最終段の状態変数から入力段の状態変数に向かって順に、それぞれの状態変数における状態数が最小化される処理が実行される。図１１に、この処理に対応する更に詳細な動作フローチャートを示す。

【０１００】まず、図１１のステップ１１０１で、最終段の状態変数において、“状態の集合”に含まれる状態のうち“動作テーブル”における値が同じ状態が、グループ化される。即ち、最終段の状態変数に対応するパイプラインユニットにおいて、そのユニットで実行される状態（命令）のうち、占有する資源、又はレジスタの読出し若しくは書込みの動作等が同じ状態がグループ化される。この結果、グループ化の対象となった状態が新たな１つの状態に置き換えられる。即ち、状態変数の“状態の集合”において、グループ化の対象となった状態が削除され、グループ化の結果を示す新たな１つの状態が登録される。また、状態変数の“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな１つの状態に関する記述が追加される。この記述は、グループ化の対象となった状態に関して共通に記述されていた内容と同じ内容を有する。

【０１０１】次に、図９のステップ９０１で決定された順序と逆の順序に従って、最終段の前段の状態変数から入力段の状態変数に向かって、ステップ１１０２～ステップ１１０４の処理が繰り返し実行される。

【０１０２】ステップ１１０２では、図９のステップ９０１で決定された順序と逆の順序に従って、最終段の状態変数の最も近くに接続され、かつ処理されていない状態変数Ｕが選択される。

【０１０３】ステップ１１０３では、状態変数Ｕの“状態の集合”に含まれる状態のうち、状態変数Ｕの“後段に接続する状態変数のリスト”に含まれる状態変数の“状態の集合”において同じグループに属し、かつ状態変数Ｕの“動作テーブル”における値が同じ状態が、グループ化される。この結果、グループ化の対象となった状態が新たな１つの状態に置き換えられる。即ち、状態変数Ｕの“状態の集合”において、グループ化の対象となった状態が削除され、グループ化の結果を示す新たな１つの状態が登録される。また、状態変数Ｕの“動作テーブル”において、グループ化の対象となった状態に関する記述が削除され、グループ化の結果を示す新たな１つの状態に関する記述が追加される。この記述は、グループ化の対象となった状態に関して記述されていた共通の内容と同じ内容を有する。上述のグループ化の処理により、グループ化が行われた状態変数の後段に接続される状態変数に対するグループ化の結果と同じ結果、又はその結果を更に細分化した結果が得られる。

【０１０４】上述したステップ１１０２と１１０３の処理が、ステップ１１０４で、処理されていない状態変数がなくなったと判定されるまで、即ち、入力段の状態変数が処理されるまで繰り返し実行される。

【０１０５】図１１の動作フローチャートで示される図９のステップ９０３の最終段側からの状態数最小化の処理により、例えば図２１に示される動作モデルM_1から図２２に示される動作モデルM_2が得られる。この動作モデルM_2において、最終段の#8の状態変数において、状態Ａ、Ｂ、Ｃ、Ｄがグループ化されることによって、それらの状態が新たな状態Ｘに置き換えられる。また、#7の状態変数において、状態Ｃ、Ｄがグループ化されることによってそれらの状態が新たな状態Ｚに置き換えられる。

【０１０６】上述した図９のステップ９０３の最終段側からの状態数最小化の処理により、それぞれの状態変数に関して、その状態変数対応するユニットから最終段のユニットまでの動作が全て同じ状態（命令）をグループ化することができる。この結果、後述する試験命令列を生成するための処理が必要とする時間を短縮することができる。

【０１０７】図９のステップ９０３の最終段側からの状態数最小化の処理が終了することにより図８のステップ８０３の状態数最小化の処理が終了する。この結果、状態数最小化部７０３は、上述の最終的に得られる動作モデル（例えば図２２に示される動作モデルM_2）を、動作モデルＭとして出力する。

【０１０８】次に、図７の入力情報保持部７０４、試験命令列列挙部７０５、及び次時刻状態計算部７０６とからなる部分は、上述した動作モデルＭと、予め設定されている試験状態の集合Ｈとから、プロセッサのパイプラインを試験するための命令列である試験命令列７０７を

生成する。

【０１０９】これらの部分が実行する動作について、図１２～図１５に示される動作フローチャートと、図２３～図２８の動作説明図に従って、詳細に説明する。まず、図１２のステップ１２０１に示されるように、図７に示される入力情報保持部７０４には、状態数最小化部７０３から出力される動作モデルMと予め設定されている試験状態の集合Hが入力され、それらが保持される。

【０１１０】動作モデルMは、前述した図１６に示される状態変数の構造体の集合として入力情報保持部７０４に保持される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMは、図２０、図２１、又は図２２に示される#1～#8の８つの状態変数に対応する８組の構造体の集合として保持される。

【０１１１】一方、試験状態の集合Hは、パイプラインハザードを発生させるような状態の集合である。例えば、理解の容易のため、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合は、初期動作モデルM_0に対応する試験状態の集合Hの例として、図２３に示される６つのケースの試験状態を設定することができる。図２３において、ケース１に示される試験状態は、#6の状態変数の状態がD、かつ#5の状態変数の状態がCとなる状態である。この状態が発生する時刻（命令サイクル）の次時刻（次の命令サイクル）には、#7の状態変数において競合即ちハザードが発生する。言い換えれば、図１８において、F_2ユニット１６０６が命令Dを実行し、かつF_1ユニット１６０５が命令Cを実行した命令サイクルの次の命令サイクルにおいて、F_3ユニット１６０７には命令Dと命令Cが同時に入力される。この場合、F_3ユニット１６０７は、命令Dと命令Cの競合を設計された仕様に従って解決することを要求される。ケース２～ケース６に示される試験状態も、ケース１の場合と同様に理解することができる。一般的には、“前段に接続する状態変数のリスト”（図１６）に複数の状態変数が登録されている状態変数を対象として、試験状態を設定することができる。

【０１１２】図７の入力情報保持部７０４、試験命令列列挙部７０５、及び次時刻状態計算部７０６とからなる部分によって自動的に生成される試験命令列７０７（図７）によって、例えば図２３に示されるような試験状態が意図的に発生させられ、この結果、そのような試験状態に対してプロセッサが正常に機能するか否かを試験することができるのである。

【０１１３】動作モデルMが、図２０に示されるような初期動作モデルM_0ではなく図２２に示されるような状態数最小化の処理が施された後の動作モデルM_2であるような場合においても、図２３の場合と同様な試験状態の集合Hを設定することができる。

【０１１４】図２３に例示したようなそれぞれの試験状態は、図７の入力情報保持部７０４においては、動作モデルMを構成するそれぞれの状態変数に試験状態を対応させたデータとして保持される。例えば、動作モデルMが図１８のパイプライン構造に対応している場合、図２３に示されるケース１の試験状態は、図２９に示されるデータ構造として入力情報保持部７０４に保持される。即ち、図２９においては、パイプラインの各ユニットに対応する#1～#8の状態変数が定義され、#6の状態変数に状態Dが、#5の状態変数に状態Cが設定される。状態が設定されていない状態変数は任意の状態を取ることができる。図２３に示されるケース２～ケース６の試験状態も、図２９に示されるケース１の場合と同様のデータ構造として入力情報保持部７０４に保持される。これらの試験状態の集合が試験状態の集合Hである。

【０１１５】上述したようにして、入力情報保持部７０４に動作モデルMと試験状態の集合Hが保持された後、ステップ１２０２では、初期状態の集合S_0として１つの初期状態が定義され、この初期状態の集合S_0が図７の試験命令列列挙部７０５へ入力される。この初期状態は、動作モデルMを構成する全ての状態変数の状態が処理する命令が無い状態となっている状態である。そして、例えば、動作モデルMが図１８に示されるパイプライン構造に対応している場合は、試験命令列列挙部７０５は、上述の初期状態を、例えば図３０に示されるデータ構造として保持する。図３０において、それぞれの状態変数に設定されている記号Oは、それぞれの状態変数に対応するパイプラインユニットにおいて処理する命令が無いことを示す状態を表している。

【０１１６】続いて、図７の試験命令列列挙部７０５は、ステップ１２０３において、初期状態の集合S_0に含まれる初期状態からの試験命令列列挙の処理を実行する。この処理では、動作モデルMが対象とする全ての試験命令列のうちから、初期状態の集合S_0に含まれる例えば図３０に示される初期状態からの基本遷移のみによって到達できる試験状態（この試験状態は、試験状態の集合Hに含まれる）を発生させる試験命令列が抽出される。ここで、基本遷移とは、動作モデルMを構成するそれぞれの状態変数における状態の変化が、パイプラインハザードを原因とするインターロックにより何れかの命令サイクルにおいて停滞するような遷移を含まない遷移をいう。

【０１１７】図１３は、図１２のステップ１２０３の処理を示す動作フローチャートである。まず、ステップ１３０１では、状態の集合Uが空集合φとされる。また、状態の集合Sに、開始状態として、図１２のステップ１２０２で定義された初期状態の集合S_0に含まれる図３０に示される初期状態が設定される。

【０１１８】続いて、ステップ１３０２～１３０５の一連の処理が繰り返し実行される。これらの一連の処理の

１回の実行は、動作モデルMに対応するパイプラインにおける１命令サイクルの時間経過に対応する。

【０１１９】上述の一連の処理のうちステップ１３０２では、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、１時刻（１命令サイクル）前に始めて到達した状態の集合Sの内容が追加される。それぞれの状態の時刻は、初期状態に対応する時刻（命令サイクル）を基準とする。そして、この処理は、状態の集合Uと状態の集合Sの論理和を演算し、その演算結果を新たな状態の集合Uとする処理として表現される。なお、状態の集合Sの初期値は、ステップ１３０１で設定される初期状態の集合S_0である。この結果得られる状態の集合Uは、後述するステップ１３０４で使用される。

【０１２０】ステップ１３０３では、動作モデルMにおいて、１時刻（１命令サイクル）前に始めて到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が計算される。そして、それぞれの状態のうち試験状態でない状態の集合が、次時刻の状態の集合S′として得られる。

【０１２１】このステップ１３０３において、基本遷移により到達する次時刻の状態が試験状態である場合には、その状態に至る命令列が図７の試験命令列列挙部７０５から試験命令列７０７として出力される。このようにして、初期状態の集合S_0に含まれる例えば図３０に示される初期状態からの基本遷移のみによって到達できる試験状態を発生させる試験命令列が抽出される。この場合、試験状態はパイプラインハザードを発生させるため、試験状態に対応する上述の状態は次時刻において基本遷移を起こすことはできない。このため、その状態は、次時刻の状態の集合S′には追加されない。

【０１２２】ステップ１３０４では、次時刻の状態の集合S′に含まれる状態のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態が、次時刻に始めて到達した状態の集合Sとされる。この処理は、状態の集合S′と、状態の集合Uの排他論理の論理積を演算し、その演算結果を新たな状態の集合Sとする処理として表現される。

【０１２３】ステップ１３０５では、次時刻に始めて到達した状態の集合Sが空集合φであるか否かが判定される。ステップ１３０５の判定がＮＯならば、次時刻に始めて到達した状態の集合Sから次時刻における基本遷移によって到達できる試験状態を検索するために、ステップ１３０２～１３０５の一連の処理が繰り返される。

【０１２４】以上の図１３のステップ１３０２～１３０５の一連の処理の繰返しによって、動作モデルMが対象とする全ての試験命令列のうちから、初期状態の集合S_0に含まれる例えば図３０に示される初期状態からの基本遷移のみによって到達できる全ての試験状態を発生させる全ての試験命令列が抽出される。

【０１２５】図１４に、図１３のステップ１３０３の処理の動作フローチャートを示す。ステップ１４０１では、次時刻の状態の集合S′が空集合φとされる。ステップ１４０２では、１時刻（１命令サイクル）前に始めて到達した状態の集合Sから、その１つの要素である状態sと、その状態sに至る命令列T_sが取り出される。今、状態の集合Sが初期状態である場合には、状態sは、例えば図３０に示される初期状態のデータとして、図７の試験命令列列挙部７０５内の特には図示しないメモリに保持されている。また、状態sに至る命令列T_sは存在しない。なぜならば、図３０に示される初期状態は、動作モデルMを構成する全ての状態変数の状態が処理する命令が無い状態となっている状態であるからである。

【０１２６】ステップ１４０３においては、図７の入力情報保持部７０４に保持されている動作モデルMを構成する全ての状態変数の“入力の集合”（図１６）に設定されている入力状態の集合が、入力状態集合Xとして定義される。なお、この入力状態集合Xには、入力される命令が無い状態も含まれる。例えば、理解の容易のため、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合は、入力状態集合Xは、命令フェッチユニットに対応する状態変数の“入力の集合”に格納されている実行可能な全ての命令の集合である。即ち、X＝｛A、B、C、D、O｝である。ここで、記号Oは、入力される命令が無いことを示す状態に対応する。動作モデルMが、図２０に示されるような初期動作モデルM_0ではなく図２２に示されるような状態数最小化の処理が施された後の動作モデルM_2であるような場合においても、動作モデルM_2を構成する全ての状態変数の“入力の集合”に設定されている入力状態の集合として、入力状態集合Xを定義することができる。

【０１２７】ステップ１４０４では、ステップ１４０３で定義された入力状態集合Xから、その１つの要素である入力状態xが取り出される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、入力状態集合X＝｛A、B、C、D｝から入力状態xとして命令Aが取り出される。

【０１２８】ステップ１４０５では、ステップ１４０２において状態の集合Sから取り出された１つの状態sと、ステップ１４０４において入力状態集合Xから取り出された１つの入力状態xに基づいて、図７の入力情報保持部７０４に保持されている動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態s′が計算される。また、状態sに至る命令列T_sに入力状態

ｘを付加した命令列が、次時刻の状態ｓ′に至る命令列$T_{s'}$として計算される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、状態ｓが図３０のデータにより示される初期状態で、入力状態ｘが命令Ａである場合には、初期動作モデルM_0である動作モデルMを構成するそれぞれの状態変数の“遷移関数”により計算される次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$は、図３１(a)のデータによって示される。この例の場合、初期動作モデルM_0である動作モデルMの状態が、そのモデルを構成する全ての状態変数において処理する命令が無い状態である初期状態から、図１９のＩＦユニット１６０１に対応する#1の状態変数が命令Ａを実行した状態に変化する。動作モデルMが、図２０に示されるような初期動作モデルM_0ではなく図２２に示されるような状態数最小化の処理が施された後の動作モデルM_2であるような場合においても、動作モデルM_2を構成するそれぞれの状態変数の“遷移関数”により、次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$を計算することができる。

【０１２９】ステップ１４０６では、ステップ１４０５で計算された次時刻の状態ｓ′が、入力情報保持部７０４に保持されている試験状態の集合Ｈに含まれるか否かが判定される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、次時刻の状態ｓ′が図３１(a)のデータによって示される状態である場合には、この状態と、図２９に示されるようなデータ形式で入力情報保持部７０４に保持されている例えば図２３に示されるそれぞれの試験状態が、一致するか否かが検査される。なお、例えば図２９に示されるケース１の試験状態（図２３参照）の場合、状態が設定されていない状態変数は任意の状態を取ることができる。

【０１３０】ステップ１４０６において、次時刻の状態ｓ′が何れの試験状態とも一致しないと判定された場合には、ステップ１４０７で、次時刻の状態の集合Ｓ′に、次時刻の状態ｓ′とその状態に至る命令列$T_{s'}$が追加される。例えば、次時刻の状態ｓ′を示す図３１(a)のデータとその状態に至る命令列$T_{s'}$のデータが、図７の試験命令列列挙部７０５内の特には図示しないメモリ内の次時刻の状態の集合Ｓ′のリストに追加される。

【０１３１】ステップ１４０９では、ステップ１４０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったか否かが判定される。ステップ１４０９の判定がＮＯなら、再びステップ１４０４が実行される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、入力状態集合Ｘ＝｛Ａ、Ｂ、Ｃ、Ｄ｝から、入力状態ｘとして、既に取り出された命令Ａ以外の命令Ｂが取り出される。

【０１３２】次に、再びステップ１４０５が実行される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、状態ｓが図３０のデータによって示される初期状態であり、入力状態ｘが命令Ｂである場合には、初期動作モデルM_0である動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって計算される次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$は、図３１(b)のデータによって示される。

【０１３３】更に、再びステップ１４０６が実行され、例えば図３１(b)のデータによって示される次時刻の状態ｓ′と、図２９に示されるようなデータ形式で入力情報保持部７０４に保持されている例えば図２３に示されるそれぞれの試験状態が、一致するか否かが検査される。

【０１３４】そして、ステップ１４０６において、再び、次時刻の状態ｓ′が何れの試験状態とも一致しないと判定された場合には、再びステップ１４０７が実行されることによって、次時刻の状態の集合Ｓ′に、次時刻の状態ｓ′を示す例えば図３１(b)のデータが追加される。

【０１３５】その後、ステップ１４０９では、再び、ステップ１４０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったか否かが判定される。上述したステップ１４０４～１４０９の一連の処理が繰り返された結果、ステップ１４０９において、ステップ１４０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったと判定された場合には、ステップ１４１０で、１時刻（１命令サイクル）前に始めて到達した状態の集合Ｓから取り出されるべき状態ｓがなくなったか否かが判定される。今、状態の集合Ｓが初期状態である場合には、状態ｓは、例えば図３０のデータで示される１つの初期状態のみであるため、ステップ１４１０の判定はＹＥＳとなる。この結果、図１３のステップ１３０３の処理が終了する。

【０１３６】図１３のステップ１３０４では、上述のようにして計算された次時刻の状態の集合Ｓ′に含まれる状態ｓ′とその状態に至る命令列$T_{s'}$のうち現在の時刻（命令サイクル）までに到達している状態の集合Ｕに含まれない状態とその状態に至る命令列が、次時刻に始めて到達した状態の集合Ｓとされる。入力状態ｘが入力される命令が無いことを示す状態であったような場合、又はパイプラインにおいてインターロックが発生することによりパイプライン全体の動作が停止したような場合においては、次時刻の状態の集合Ｓ′に含まれる状態が現在の時刻（命令サイクル）までに到達している状態の集

合Ｕに含まれる結果となる可能性がある。このため、ステップ１３０４の処理が必要となる。

【０１３７】その後、次時刻（次の命令サイクル）に対応して、再び図１３のステップ１３０３に対応する図１４の動作フローチャートが実行される。そして、ステップ１４０２において、１時刻（１命令サイクル）前に始めて到達した状態の集合Ｓから、その１つの要素である状態ｓとその状態ｓに至る命令列T_sとして、例えば図３１(a)に示されるデータが取り出される。また、ステップ１４０４において、ステップ１４０３で定義された入力状態集合Ｘから、その１つの要素である入力状態ｘとして、例えば命令Ａがが取り出される。この例でステップ１４０５が実行されると、初期動作モデルM_0である動作モデルＭを構成するそれぞれの状態変数の“遷移関数”により計算される次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$は、図３２(a)のデータによって示される。この例の場合、初期動作モデルM_0である動作モデルＭの状態が、図３１(a)のデータで示されるように図１９のＩＦユニット１６０１に対応する#1の状態変数が命令Ａを実行した状態から、図３２(a)のデータで示されるように図１９のＩＤユニット１６０２に対応する#2の状態変数がＩＦユニット１６０１から出力された命令Ａを実行し、かつＩＦユニット１６０１に対応する#1の状態変数が新たな命令Ａを実行した状態に変化する。

【０１３８】この例においてステップ１４０６が実行されると、例えば図３２(a)のデータによって示される次時刻の状態ｓ′と、図２９に示されるようなデータ形式で入力情報保持部７０４に保持されている例えば図２３に示されるそれぞれの試験状態が、一致するか否かが検査される。そして、ステップ１４０６において、次時刻の状態ｓ′が何れの試験状態とも一致しないと判定された場合には、ステップ１４０７が実行されることによって、次時刻の状態の集合Ｓ′に、次時刻の状態ｓ′とその状態に至る命令列$T_{s'}$を示す、例えば図３２(a)のデータが追加される。

【０１３９】その後、ステップ１４０９では、ステップ１４０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったか否かが判定される。ステップ１４０９の判定がＮＯなら、再びステップ１４０４が実行される。上述の例において再びステップ１４０４が実行されると、入力状態集合Ｘから、入力状態ｘとして、既に取り出された命令Ａ以外の命令Ｂが取り出される。この例でステップ１４０５が実行されると、初期動作モデルM_0である動作モデルＭを構成するそれぞれの状態変数の“遷移関数”により計算される次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$は、図３２(b)のデータによって示される。

【０１４０】この例において再びステップ１４０６が実行されると、例えば図３２(b)のデータによって示される次時刻の状態ｓ′と、図２９に示されるようなデータ形式で入力情報保持部７０４に保持されている例えば図２３に示されるそれぞれの試験状態が、一致するか否かが検査される。そして、ステップ１４０６において、次時刻の状態ｓ′が何れの試験状態とも一致しないと判定された場合には、ステップ１４０７が実行されることによって、次時刻の状態の集合Ｓ′に、次時刻の状態ｓ′とその状態に至る命令列$T_{s'}$を示す、例えば図３２(b)のデータが追加される。

【０１４１】上述したステップ１４０４～１４０９の一連の処理が繰り返された結果、ステップ１４０９において、ステップ１４０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったと判定された場合には、ステップ１４１０で、１時刻（１命令サイクル）前に始めて到達した状態の集合Ｓから取り出されるべき状態ｓがなくなったか否かが判定される。上述の例においてステップ１４１０が実行されると、状態の集合Ｓには、既に処理された図３１(a)に示される状態のほかに、例えば図３１(b)に示される状態なども存在する。従って、ステップ１４１０の判定はＮＯとなる。

【０１４２】ステップ１４１０の判定がＮＯなら、再びステップ１４０２～１４０９が実行される。この結果、状態ｓとして取り出される例えば図３１(b)に示される状態と、入力状態集合Ｘに含まれる全ての入力状態ｘとから、例えば図３３(a)及び(b)に示される次時刻の状態ｓ′とその状態に至る命令列$T_{s'}$が計算される。そして、それらの状態ｓ′は試験状態の集合Ｈには含まれないため、それらの状態ｓ′とその状態に至る命令列$T_{s'}$が次時刻の状態の集合Ｓ′に追加される。

【０１４３】以上のようにして、時刻（命令サイクル）の経過に従って現在の時刻までに到達している状態の集合Ｕが計算されてゆく。そして、図１４のステップ１４０５が実行された時点において、次時刻の状態ｓ′が例えば図３４に示されるデータとして計算された場合、この状態は図２９及び図２３に示されるケース１の試験状態と一致する。この結果、ステップ１４０６の判定がＹＥＳとなり、ステップ１４０８が実行される。このステップ１４０８では、例えば図３４に示されるように、次時刻の状態ｓ′に至る命令列$T_{s'}$＝｛Ｄ、Ｃ、Ａ、Ａ｝が、図７の試験命令列列挙部７０５から試験命令列７０７として出力される。この場合、試験状態はパイプラインハザードを発生させるため、試験状態に対応する図３４に示される次時刻の状態ｓ′は更にその次の時刻において基本遷移を起こすことはできない。このため、図３４に示されるその状態とその状態に至る命令列は、次時刻の状態の集合Ｓ′には追加されない。

【０１４４】ステップ１４０８の処理の後は、ステップ１４０９が実行され、次時刻における別の状態ｓ′が探索される。ステップ１４０２～１４１０の一連の処理が繰り返され、状態の集合Ｓから取り出されるべき状態ｓがなくなったと判定されると、図１３のステップ１３０

３の処理が終了する。

【０１４５】上述の図１４の動作フローチャートの処理を含む図１３のステップ１３０２～１３０５の一連の処理の繰返しにより示される図１２のステップ１２０３の処理が図７の試験命令列列挙部７０５において実行されることにより、動作モデルMが対象とする全ての試験命令列のうちから、初期状態の集合S_0に含まれる、例えば図３０に示される初期状態からの基本遷移のみにより到達できる全ての試験状態を発生させる全ての試験命令列７０７が抽出される。図２４(a)～(d)に、図３０に示される初期状態からの基本遷移のみによって到達できる図２３に示されるケース１の試験状態を発生させる試験命令列７０７の例を示す。図２４(a)のケース１－１において示されている試験命令列＝｛D、C、A、A｝が、図３４の例に対応する。

【０１４６】試験状態の集合Hに図２３に示される試験状態が含まれる場合には、図２３のケース２の試験状態も図３０に示される初期状態からの基本遷移のみによって到達することができる。そして、特には図示しないが、この試験状態を発生させる試験命令列７０７も、図７の試験命令列列挙部７０５が図１２のステップ１２０３の処理を実行することにより、抽出することができる。

【０１４７】図７の試験命令列列挙部７０５が上述した図１２のステップ１２０３の処理を実行することにより抽出される試験命令列７０７は、入力情報保持部７０４に入力される試験状態の集合Hに含まれる例えば図２３として示される全ての試験状態を発生させるものではなく、初期状態からの基本遷移のみによって到達できる試験状態を発生させるものである。そこで、図７の試験命令列列挙部７０５と次時刻状態計算部７０６は、更に、図１２のステップ１２０４～１２１１を実行することにより、初期状態からの基本遷移のみにより到達できる試験状態以外の試験状態を発生させる試験命令列７０７を抽出する。

【０１４８】具体的には、まず、図７の次時刻状態計算部７０６が、図７の試験命令列列挙部７０５が図１２のステップ１２０３の処理を実行することにより抽出したそれぞれの試験命令列７０７に対応する試験状態の発生時刻（発生命令サイクル）の次時刻（次の命令サイクル）の状態を、動作モデルMを構成するそれぞれの状態変数の“遷移関数”として設定されている設計仕様に従って計算する。この場合に、動作モデルMを構成する状態変数のうち上述の試験状態に関係する状態変数においては、その状態変数の“遷移関数”として設定されている設計仕様に従って、上述の試験状態によって引き起こされるパイプラインハザードが解決されるようにその状態変数自身の状態が変化させられる。

【０１４９】その後、図７の試験命令列列挙部７０５は、次時刻状態計算部７０６によって計算された新たな状態からの基本遷移のみにより到達できる試験状態を発生させる試験命令列７０７を抽出する。

【０１５０】上述の動作を実現する図１２のステップ１２０４～１２１１の処理の詳細について、以下に説明する。なお、ステップ１２０４～１２０８、及びステップ１２１０、１２１１は、図７の次時刻状態計算部７０６が実行し、ステップ１２０９は、図７の試験命令列列挙部７０５が実行する。

【０１５１】まず、ステップ１２０４において、図７の試験命令列列挙部７０５が前述した図１２のステップ１２０３の処理を実行することによって抽出した試験命令列７０７に対応する試験状態が、試験状態の集合H_1に含ませられる。そして、入力情報保持部７０４に保持されている試験状態の集合Hに含まれ、かつ、試験状態の集合H_1に含まれない試験状態が、試験状態の集合～H_1に含ませられる。なお、記号“～H_1”は、図１２のステップ１２０４中では、記号“H_1”の上に記号“～”が付加された記号として表現されている。この試験状態の集合～H_1は、後述するステップ１２１０で使用される。

【０１５２】次に、ステップ１２０５で、繰返しの回数を示す変数ｉの値が、１に初期設定される。その後、ステップ１２１１で変数ｉの値が＋１ずつインクリメントされながら、ステップ１２０６～１２１１の一連の処理が繰り返し実行される。

【０１５３】まず、ステップ１２０６において試験状態の集合H_iに試験状態が１つも含まれなくなったと判定された場合、或いは、ステップ１２０７において試験状態の集合～H_{i+1}に試験状態が１つも含まれなくなったと判定された場合には、試験命令列７０７の抽出処理が終了する。ここで、H_iの初期値はH_1、～H_iの初期値は～H_1である。これらは、前述したステップ１２０４において計算される。その後、H_i及び～H_iは、ステップ１２１０において計算される。

【０１５４】次に、図７の次時刻状態計算部７０６は、ステップ１２０８において、試験状態の集合H_iの次時刻（次の命令サイクル）の状態の集合K_iを計算する。図１５に、図１２のステップ１２０８の処理の動作フローチャートを示す。

【０１５５】ステップ１５０１では、次時刻の状態の集合K_iが空集合φとされる。ステップ１５０２では、処理対象である試験状態の集合H_iから、その１つの要素である状態ｈと、その状態ｈに至る命令列T_hが取り出される。例えば、図１８に示されるパイプライン構造に対応する動作モデルMが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、図１２における変数ｉの値が１である場合には、試験状態の集合H_i＝H_1に含まれる状態ｈに至る命令列T_hの一部には、図２４(a)～(d)に示されるように、前述した図１２のステップ１

２０３において抽出された８０種類の試験命令列７０７が含まれる。また、試験状態の集合H_i＝H_1の要素の一部には、状態ｈとして、図２４(a)～(d)に示されるように、上述の８０種類の試験命令列７０７に対応する８０種類の試験状態が含まれる。これらの試験状態は、図２３のケース１の試験状態に対応する。更に、特には図示しないが、上述の例において、試験状態の集合H_iに含まれる状態ｈとその状態ｈに至る命令列T_hとして、図２３のケース２の試験状態に対応するものも存在する。

【０１５６】ステップ１５０３においては、前述した図１４のステップ１４０３の場合と同様にして、図７の入力情報保持部７０４に保持されている動作モデルＭを構成する全ての状態変数の“入力の集合”（図１６）に設定されている入力状態の集合が、入力状態集合Ｘとして定義される。

【０１５７】ステップ１５０４では、前述した図１４のステップ１４０４の場合と同様にして、ステップ１５０３で定義された入力状態集合Ｘから、その１つの要素である入力状態ｘが取り出される。

【０１５８】ステップ１５０５では、ステップ１５０２において試験状態の集合H_iから取り出された１つの状態ｈと、ステップ１５０４において入力状態集合Ｘから取り出された１つの入力状態ｘに基づいて、図７の入力情報保持部７０４に保持されている動作モデルＭを構成するそれぞれの状態変数に設計仕様として設定されている“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態ｋが計算される。また、状態ｈに至る命令列T_hに入力状態ｘを付加した命令列が、次時刻の状態ｋに至る命令列T_kとして計算される。この場合、動作モデルＭを構成する状態変数のうち上述の試験状態ｈに関係する状態変数においては、その状態変数の“遷移関数”として設定されている設計仕様に従って、上述の試験状態ｈによって引き起こされるパイプラインハザードが解決されるようにその状態変数自身の状態が変化させられる。例えば、図１８に示されるパイプライン構造に対応する動作モデルＭが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、図１２における変数ｉの値が１であって、かつ、状態ｈが図２４(a)～(d)の何れかの例に含まれる場合は、図１９のF_3ユニット１６０７に対応する#7の状態変数の“遷移関数”が、図１９のF_1ユニット１６０５に対応する#5の状態変数から#7の状態変数に命令Ｃが入力されると同時に図１９のF_2ユニット１６０６に対応する#6の状態変数から#7の状態変数に命令Ｄが入力されるというパイプラインハザードを解決するように、#7の状態変数自身の状態を変化させる。より具体的には、例えば上記２つの命令のうち先にパイプラインでの実行が開始された命令（図２４の例では命令Ｄ）を先に実行するという関数規則を有する“遷移関数”が適用される。動作モデルＭが、図２０に示されるような初期動作モデルM_0でなく図２２に示されるような状態数最小化の処理が施された後の動作モデルM_2であるような場合においても、動作モデルM_2を構成するそれぞれの状態変数の“遷移関数”により、次時刻の状態ｋと、その状態に至る命令列T_kを計算することができる。

【０１５９】ステップ１５０６では、次時刻の状態の集合K_iに、次時刻の状態ｋとその状態に至る命令列T_kが追加される。これらのデータは、図７の次時刻状態計算部７０６においては、例えば前述した図３１(a)のデータの形式と同様の形式で処理され、これらが次時刻状態計算部７０６内の特には図示しないメモリ内の次時刻の状態の集合K_iのリストに追加される。

【０１６０】ステップ１５０７では、ステップ１５０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったか否かが判定される。ステップ１５０７の判定がＮＯなら、再びステップ１５０４～１５０６の一連の処理が実行される。

【０１６１】上述したステップ１５０４～１５０７の一連の処理が繰り返された結果、ステップ１５０７において、ステップ１５０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったと判定された場合には、ステップ１５０８で、処理対象である試験状態の集合H_iから取り出されるべき状態ｈがなくなったか否かが判定される。

【０１６２】ステップ１５０８の判定がＮＯなら、再びステップ１５０２～１５０７が実行される。この結果、状態ｈとして取り出される例えば図２４(a)～(d)に示されるそれぞれの試験状態と、入力状態集合Ｘに含まれる全ての入力状態ｘとから、次時刻の状態ｋとその状態に至る命令列T_kが計算される。そして、それらの状態ｋが次時刻の状態の集合K_iに追加される。

【０１６３】処理対象である試験状態の集合H_iから取り出されるべき状態ｈがなくなり、ステップ１５０８の判定がＹＥＳとなると、図１２のステップ１２０８の処理が終了する。

【０１６４】ここで、例えば、図１８に示されるパイプライン構造に対応する動作モデルＭが前述した状態数最小化の処理が施されていない図２０に示される初期動作モデルM_0であると仮定した場合において、図１２における変数ｉの値が１であり、かつ、試験状態の集合H_i＝H_1の要素の一部に図２４(a)～(d)に示される試験状態が含まれる場合は、上記ステップ１２０８の処理が実行される結果、図２４(a)～(d)に示される試験状態が発生した時刻の次時刻の状態ｋと、その状態ｋに至る命令列T_kとして、図２５(a)～(d)に示されるものが計算される。なお、図２４(a)～(d)に示される各ケースから図２５(a)～(d)に示される各ケースへの遷移関係は、図２６に示される。

【０１６５】次に、図７の試験命令列列挙部７０５は、図１２のステップ１２０９で、次時刻状態計算部７０６が図１２のステップ１２０８において計算した新たな状態の集合K_iに含まれるそれぞれの状態からの試験命令列列挙の処理を実行する。この処理では、動作モデルMが対象とする全ての試験命令列のうちから、状態の集合K_iに含まれるそれぞれの状態からの基本遷移のみによって到達できる試験状態を発生させる試験命令列７０７が抽出される。図１２のステップ１２０９の処理は、図１２のステップ１２０３の処理と全く同様の、図１３及び図１４の動作フローチャートにより示される。この処理の結果、例えば、図２５(a)～(d)に示される状態からの基本遷移のみによって到達できる図２３に示されるケース５の試験状態を発生させる試験命令列７０７の例として、図２７(a)～(d)に示されるものが計算される。なお、図２５(a)～(d)に示される各ケースから図２７(a)～(d)に示される各ケースへの遷移関係は、図２８に示される。

【０１６６】以上のようにして、図７の試験命令列列挙部７０５から新たな試験命令列７０７が出力される。続いて、図７の次時刻状態計算部７０６は、図１２のステップ１２１０において、図７の試験命令列列挙部７０５が上述した図１２のステップ１２０９の処理を実行することにより抽出した試験命令列７０７に対応する試験状態が、試験状態の集合H_{i+1}に含ませられる。そして、今まで抽出されていない試験状態の集合$\sim H_i$に含まれ、かつ、新たに抽出された試験状態の集合H_{i+1}に含まれない試験状態が、試験状態の集合$\sim H_{i+1}$に含ませられる。なお、記号“$\sim H_{i+1}$”は、図１２のステップ１２１０中では、記号“H_{i+1}”の上に記号“～”が付加された記号として表現されている。

【０１６７】その後、ステップ１２１１で変数ｉの値が＋１インクリメントされた後に、ステップ１２０６～１２１１の処理が繰り返し実行される。以上の一連の処理が、ステップ１２０６又は１２０７での判定がＹＥＳとなるまで繰り返し実行されることにより、試験命令列列挙部７０５は、入力情報保持部７０４に保持されている試験状態の集合Ｈに含まれる全ての試験状態に対応する試験命令列７０７を出力することができる。

＜本発明の第２の実施例＞次に、本発明の第２の実施例について説明する。

【０１６８】第２の実施例では、動作モデルに対して試験項目を指定するための入力情報として、第１の実施例におけるような或る１つの時刻における試験状態の集合ではなく、状態評価関数を指定できる点が特徴である。

【０１６９】図３５は、本発明の第２の実施例の全体構成図である。この構成は、図７に示される第１の実施例に対応する構成のうち、７０４～７０７の各部分を置き換えて得られる構成である。従って、第２の実施例において、動作モデルMを生成する部分の構成及び機能は、特には図示しないが、第１の実施例における図７の７０１～７０３によって示される構成及び機能と同一である。

【０１７０】まず、入力情報保持部３５０１は、動作モデルMとそのモデルに対する状態評価関数とからなる入力情報を保持する。到達可能状態列挙部３５０２は、入力情報保持部３５０１に保持された動作モデルMにおいて、初期状態から到達することのできる全ての状態を列挙する。このとき同時に、各状態に対する状態評価関数も計算される。

【０１７１】試験命令列列挙部３５０３は、到達可能状態列挙部３５０２によって列挙された、初期状態から到達可能な各状態から、それら各状態について計算される状態評価関数の値に基づいて、所定の状態を選択し、その状態に至る試験命令列３５０４を列挙する。

【０１７２】以上の構成を有する第２の実施例の動作について、図３６及び図３７の動作フローチャートに従って、詳細に説明する。図３６は、第２の実施例の全体動作フローチャートである。

【０１７３】まず、図３６のステップ３６０１に示されるように、図３５に示される入力情報保持部３５０１には、第１の実施例における図７に示される状態数最小化部７０３から出力される動作モデルMと予め設定されている状態評価関数が入力されて、それらが保持される。

【０１７４】動作モデルMは、第１の実施例の場合と同様に、前述した図１６に示される状態変数の構造体の集合として入力情報保持部３５０１に保持される。一方、状態評価関数は、動作モデルMの状態が入力されるとその評価値を出力する関数であり、動作モデルMによって示されるパイプライン機構の設計誤りが検出されやすい状態（パイプライン機構に負荷を与える状態）ほど大きな値を出力するように定義される関数である。

【０１７５】具体的には、例えばパイプラインプロセッサでは、トラップを発生させた命令より前に実行された命令はその実行を完了し、トラップを発生させた命令より後に実行されるべき命令は実行されないままであることを検証しなければならない場合がある。そのためには、パイプラインにできる限り多くの命令が詰った状態かつトラップを発生させる状態が試験項目となるように、状態評価関数が定義されればよい。そのような状態評価関数は、例えば次式によって定義される。

【０１７６】

【数１】$V(q) = N_{pipe}(q) \cdot \delta_{trap}(q)$

ここで、ｑは、動作モデルMの状態、$N_{pipe}(q)$は、状態ｑにおいてパイプライン中に存在する命令の数、$\delta_{trap}(q)$は、トラップを発生する状態のときに値１をとり、その他の場合に値０をとるデルタ関数である。

【０１７７】上述の数１式で示されるような状態評価関数が、所定のデータ形式で入力情報保持部３５０１に保持される。上述したようにして、入力情報保持部３５０

１に動作モデルMと状態評価関数が保持された後、図３５の到達可能状態列挙部３５０２によって、ステップ３６０２～３６０６の処理が実行される。これらの処理により、動作モデルMにおいて、その初期状態から到達することのできる全ての状態が列挙される。

【０１７８】まず、ステップ３６０２では、初期状態から到達することのできる状態の集合Uが空集合φとされる。また、状態の集合Sに、初期状態が設定される。この初期状態は、第１の実施例の場合と同様、動作モデルMを構成する全ての状態変数の状態が処理する命令が無い状態となっている状態である。そして、例えば、動作モデルMが図１８に示されるパイプライン構造に対応している場合は、上述の初期状態は、例えば図３０に示されるデータ構造として保持される。

【０１７９】続いて、ステップ３６０３～３６０６の一連の処理が繰り返し実行される。これらの一連の処理は、第１の実施例における図１３に示される動作フローチャートのステップ１３０２～１３０５の一連の処理に対応しており、これらの一連の処理の１回の実行は、動作モデルMに対応するパイプラインにおける１命令サイクルの時間経過に対応する。

【０１８０】上述の一連の処理のうちステップ３６０３では、第１の実施例における図１３のステップ１３０２と同様に、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、１時刻（１命令サイクル）前に始めて到達した状態の集合Sの内容が追加される。なお、状態の集合Sの初期値は、ステップ３６０２で設定される初期状態の集合である。

【０１８１】ステップ３６０４では、動作モデルMにおいて、１時刻（１命令サイクル）前に始めて到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が、次時刻の状態の集合S′として計算される。

【０１８２】ステップ３６０５においては、第１の実施例における図１３のステップ１３０４と同様に、次時刻の状態の集合S′に含まれる状態のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態が、次時刻に始めて到達した状態の集合Sとされる。

【０１８３】ステップ３６０６では、第１の実施例における図１３のステップ１３０５と同様に、次時刻に始めて到達した状態の集合Sが空集合φであるか否かが判定される。

【０１８４】ステップ３６０６の判定がＮＯならば、次時刻に始めて到達した状態の集合Sから更に次時刻における基本遷移によって到達できる状態を検索するために、ステップ３６０３～３６０６の一連の処理が繰り返される。

【０１８５】以上の図３６のステップ３６０３～３６０６の一連の処理の繰返しによって、例えば図３０に示される初期状態からの基本遷移のみによって到達できる全ての状態が列挙される。

【０１８６】図３７に、図３６のステップ３６０４の処理の動作フローチャートを示す。この動作フローチャートは、第１の実施例における図１４の動作フローチャートに対応している。

【０１８７】ステップ３７０１では、次時刻の状態の集合S′が空集合φとされる。ステップ３７０２では、１時刻（１命令サイクル）前に始めて到達した状態の集合Sから、その１つの要素である状態ｓと、その状態ｓに至る命令列T_sが取り出される。第１の実施例における図１４のステップ１４０２と同様に、今、状態の集合Sが初期状態である場合には、状態ｓは、例えば図３０に示される初期状態のデータとして、図３５の到達可能状態列挙部３５０２内の特には図示しないメモリに保持されている。また、状態ｓに至る命令列T_sは存在しない。

【０１８８】ステップ３７０３においては、第１の実施例における図１４のステップ１４０３と同様に、図７の入力情報保持部７０４に保持されている動作モデルMを構成する全ての状態変数の“入力の集合”（図１６）に設定されている入力状態の集合が、入力状態集合Xとして定義される。なお、この入力状態集合Xには、入力される命令が無い状態も含まれる。

【０１８９】ステップ３７０５においては、第１の実施例における図１４のステップ１４０４と同様に、ステップ３７０２において状態の集合Sから取り出された１つの状態ｓと、ステップ３７０４において入力状態集合Xから取り出された１つの入力状態ｘに基づいて、図７の入力情報保持部７０４に保持されている動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態ｓ′が計算される。また、状態ｓに至る命令列T_sに入力状態ｘを付加した命令列が、次時刻の状態ｓ′に至る命令列$T_{s'}$として計算される。

【０１９０】ステップ３７０６では、動作モデルMの次時刻の状態ｓ′を入力として、状態評価関数Ｖ（ｓ′）が計算される。具体的には、状態評価関数Ｖ（ｓ′）が例えば前述の数１式で定義される場合には、まず、$\delta_{trap}(q)$の値として、状態ｓ′がトラップを発生する状態のときには１、その他の場合には０が設定される。次に、第１の実施例の場合と同様の例えば図３１に示されるようなデータ形式を有する状態ｓ′について、何れか命令に対応する記号Ｏ以外の記号が設定されている状態変数の数が計数されることにより、状態ｓ′においてパイプライン中に存在する命令の数N_{pipe}（ｓ′）が算出される。その後、数１式に基づいて、状態評価関数Ｖ（ｓ′）が計算される。

【０１９１】ステップ３７０７では、次時刻の状態の集合S′に、次時刻の状態ｓ′と、その状態に至る命令列

$T_{s'}$と、その状態における状態評価関数Ｖ（ｓ′）の計算結果が追加される。

【０１９２】ステップ３７０８では、第１の実施例における図１４のステップ１４０９と同様に、ステップ３７０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったか否かが判定される。

【０１９３】ステップ３７０８の判定がＮＯなら、再びステップ３７０４～３７０８が実行されることにより、新たな入力状態ｘを入力とする、状態ｓの次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$と、その状態における状態評価関数Ｖ（ｓ′）とが算出され、それらが次時刻の状態の集合Ｓ′に追加される。

【０１９４】上述したステップ３７０４～３７０８の一連の処理が繰り返された結果、ステップ３７０８において、ステップ３７０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったと判定された場合には、第１の実施例における図１４のステップ１４１０と同様に、ステップ３７０９で、１時刻（１命令サイクル）前に始めて到達した状態の集合Ｓから取り出されるべき状態ｓがなくなったか否かが判定される。

【０１９５】ステップ３７０９の判定がＮＯなら、再びステップ３７０２～３７０８が実行される。この結果、新たな状態ｓと、入力状態集合Ｘに含まれる全ての入力状態ｘとから、状態ｓの次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$と、その状態における状態評価関数Ｖ（ｓ′）とが算出され、それらが次時刻の状態の集合Ｓ′に追加される。

【０１９６】ステップ３７０２～３７０９の一連の処理が繰り返され、状態の集合Ｓから取り出されるべき状態ｓがなくなったと判定されると、ステップ３７０９の判定がＹＥＳとなって、図３６ステップ３６０４の処理が終了する。

【０１９７】以上に示されるステップ３６０２～３６０６の処理が図３５の到達可能状態列挙部３５０２により実行されることによって、動作モデルＭにおいて、その初期状態から到達することのできる全ての状態が列挙される。

【０１９８】その後、図３６のステップ３６０７の処理が図３５の試験命令列列挙部３５０３によって実行される。このステップでは、上述の一連の処理によって初期状態から到達することのできる状態の集合Ｕに登録されている、状態ｓと、その状態に至る命令列T_sと、その状態における状態評価関数Ｖ（ｓ）の複数の組合わせにおいて、状態評価関数Ｖ（ｓ）の値が大きい順に、所定個の状態が選択され、その状態に至る命令列が上記集合Ｕから取り出され、それが試験命令列３５０４（図３５）として出力される。

【０１９９】上述の第２の実施例により、定性的に表現されるような試験項目を状態評価関数の形式で指定することができ、それに対応する試験命令列３５０４を、自動的かつ効率的に生成することができる。

＜本発明の第３の実施例＞次に、本発明の第３の実施例について説明する。

【０２００】第３の実施例では、動作モデルに対して試験項目を指定するための入力情報として、状態時系列を指定できる点が特徴である。本発明の第３の実施例の全体構成図は、第２の実施例の場合と同じ図３５によって示される。

【０２０１】第３の実施例においては、入力情報保持部３５０１は、動作モデルＭとそのモデルに対する状態時系列の集合とからなる入力情報を保持する。到達可能状態列挙部３５０２は、入力情報保持部３５０１に保持された動作モデルＭにおいて、初期状態から到達することのできる全ての状態を列挙する。

【０２０２】試験命令列列挙部３５０３は、到達可能状態列挙部３５０２によって列挙された、初期状態から到達可能な各状態から、それら各状態によって構成される状態時系列のうち、入力情報保持部３５０１に記憶されている状態時系列の集合内の１つの状態時系列と一致するものを選択し、その状態時系列を実現する試験命令列３５０４を列挙する。

【０２０３】以上の構成を有する第３の実施例の動作について、図３８～図４０の動作フローチャートに従って、詳細に説明する。図３８は、第３の実施例の全体動作フローチャートである。

【０２０４】まず、図３８のステップ３８０１に示されるように、図３５に示される入力情報保持部３５０１には、第１の実施例における図７に示される状態数最小化部７０３から出力される動作モデルＭと予め設定されている状態時系列が入力され、それらが保持される。

【０２０５】動作モデルＭは、第１、第２の実施例の場合と同様に、前述した図１６に示される状態変数の構造体の集合として入力情報保持部３５０１に保持される。一方、状態時系列は、動作モデルＭが実現する或る任意の期間の状態の時系列であり、試験項目に対応するパイプライン機構の動作条件となる一定期間の動作モデルＭの状態の時系列を規定する。なお、一般に１つの試験項目に対して複数の状態時系列が存在し得る。

【０２０６】具体的には、或る種のパイプラインプロセッサにおいて、命令の実行結果の書込みユニット（例えば、第１の実施例に関する図１８のパイプライン構成におけるライトバックユニット１８０８）とレジスタファイルとの間にバッファが設けられ、このバッファによって実際のライトバック動作がプログラム上のステップの順番に対応するように修正されることにより、正確なトラップが実現されるように設計される場合がある。このようなパイプライン機構を検証するためには、数命令サイクルの間命令の完了順序が入れ替わる状態を発生させ、バッファにデータを保持させた後に、トラップを発生させるような試験命令列を生成する必要がある。第３

の実施例では、このような要求を満たす状態時系列の集合が、入力情報保持部３５０１に入力され保持される。

【０２０７】上述したようにして、入力情報保持部３５０１に動作モデルMと状態時系列の集合が保持された後、図３５の到達可能状態列挙部３５０２によって、ステップ３８０２～３８０６の処理が実行される。これらの処理により、動作モデルMにおいて、その初期状態から到達することのできる全ての状態が列挙される。これらの処理は、第２の実施例における図３６のステップ３６０２～３６０６の一連の処理に対応している。

【０２０８】まず、ステップ３８０２では、第２の実施例における図３６のステップ３６０２と同様に、初期状態から到達することのできる状態の集合Uが空集合φとされる。また、状態の集合Sに、初期状態が設定される。

【０２０９】続いて、ステップ３８０３～３８０６の一連の処理が繰り返し実行される。これらの一連の処理は、第１の実施例における図１３のステップ１３０２～１３０５の一連の処理又は第２の実施例における図３６のステップ３６０３～３６０６の一連の処理に対応している。

【０２１０】上述の一連の処理のうちステップ３８０３では、第１の実施例における図１３のステップ１３０２と同様に、動作モデルMにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Uに、１時刻（１命令サイクル）前に到達した状態の集合Sの内容が追加される。なお、状態の集合Sの初期値は、ステップ３８０２で設定される初期状態の集合である。

【０２１１】ステップ３８０４においては、動作モデルMにおいて、１時刻（１命令サイクル）前に到達した状態の集合Sに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が、次時刻の状態の集合S′として計算される。

【０２１２】ステップ３８０５においては、次時刻の状態の集合S′に含まれる状態が、次時刻に到達した状態の集合Sとされる。この処理は、第１の実施例における図１３のステップ１３０４及び第２の実施例における図３６のステップ３６０５とは異なって、次時刻の状態の集合S′に含まれる状態が、現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれていても、その状態は次時刻に到達した状態の集合Sとされる。これは、第３の実施例では、ある１時刻における状態だけではなく状態の時系列が問題となるため、状態の遷移順序を保存する必要があるためである。従って、状態の集合Uには、複数の同じ到達状態に対して異なるラベルが付与され得る。

【０２１３】一方において、ステップ３８０５では、次時刻の状態の集合S′に含まれる状態のうち現在の時刻（命令サイクル）までに到達している状態の集合Uに含まれない状態も状態S″として保存される。

【０２１４】そして、続くステップ３８０６で、この状態S″が空集合φであるか否かが判定される。即ち、どのように状態が遷移しても新しい状態が１つも発生しなくなりこのステップ３８０６の判定がＹＥＳとなった時点で、到達可能状態列挙部３５０２の動作が終了することになる。

【０２１５】ステップ３８０６の判定がＮＯならば、次時刻に到達した状態の集合Sから更に次時刻における基本遷移によって到達できる状態を検索するために、ステップ３８０３～３８０６の一連の処理が繰り返される。

【０２１６】以上の図３８のステップ３８０３～３８０６の一連の処理の繰返しによって、例えば図３０に示される初期状態からの基本遷移のみによって到達できる全ての状態が列挙される。

【０２１７】図３９に、図３８のステップ３８０４の処理の動作フローチャートを示す。この動作フローチャートは、第１の実施例における図１４の動作フローチャート又は第２の実施例における図３７の動作フローチャートに対応している。

【０２１８】ステップ３９０１では、次時刻の状態の集合S′が空集合φとされる。ステップ３９０２では、第１の実施例における図１４のステップ１４０２と同様に、１時刻（１命令サイクル）前に到達した状態の集合Sから、その１つの要素である状態ｓと、その状態ｓに至る命令列T_sが取り出される。

【０２１９】ステップ３９０３においては、第１の実施例における図１４のステップ１４０３と同様に、図７の入力情報保持部７０４に保持されている動作モデルMを構成する全ての状態変数の“入力の集合”（図１６）に設定されている入力状態の集合が、入力状態集合Xとして定義される。

【０２２０】ステップ３９０５においては、第１の実施例における図１４のステップ１４０４と同様に、ステップ３９０２において状態の集合Sから取り出された１つの状態ｓと、ステップ３９０４において入力状態集合Xから取り出された１つの入力状態ｘに基づいて、図７の入力情報保持部７０４に保持されている動作モデルMを構成するそれぞれの状態変数の“遷移関数”によって、それぞれの状態変数の次時刻（次の命令サイクル）の状態ｓ′が計算される。また、状態ｓに至る命令列T_sに入力状態ｘを付加した命令列が、次時刻の状態ｓ′に至る命令列$T_{s'}$として計算される。

【０２２１】ステップ３９０６では、次時刻の状態の集合S′に、次時刻の状態ｓ′と、その状態に至る命令列$T_{s'}$と、その状態の１時刻前の状態である現在時刻の状態ｓが追加される。現在時刻の状態ｓが記憶されるのは、後述する図３８のステップ３８０７の処理において、各到達状態から順次前の状態にバックトラックできるようにするためである。

【０２２２】ステップ３９０７では、第１の実施例にお

ける図１４のステップ１４０９と同様に、ステップ３９０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったか否かが判定される。

【０２２３】ステップ３９０７の判定がＮＯなら、再びステップ３９０４～３９０７が実行されることにより、新たな入力状態ｘを入力とする、状態ｓの次時刻の状態ｓ′と、その状態に至る命令列T_sと、その状態の１時刻前の状態である現在時刻の状態ｓとが算出され、それらが次時刻の状態の集合Ｓ′に追加される。

【０２２４】上述したステップ３９０４～３９０７の一連の処理が繰り返された結果、ステップ３９０７において、ステップ３９０３で定義された入力状態集合Ｘから取り出されるべき入力状態ｘがなくなったと判定された場合には、第１の実施例における図１４のステップ１４１０と同様に、ステップ３９０８で、１時刻（１命令サイクル）前に到達した状態の集合Ｓから取り出されるべき状態ｓがなくなったか否かが判定される。

【０２２５】ステップ３９０８の判定がＮＯなら、再びステップ３９０２～３９０７が実行される。この結果、新たな状態ｓと、入力状態集合Ｘに含まれる全ての入力状態ｘとから、状態ｓの次時刻の状態ｓ′と、その状態に至る命令列T_sと、その状態の１時刻前の状態である現在時刻の状態ｓとが算出され、それらが次時刻の状態の集合Ｓ′に追加される。

【０２２６】ステップ３９０２～３９０８の一連の処理が繰り返され、状態の集合Ｓから取り出されるべき状態ｓがなくなったと判定されると、ステップ３９０８の判定がＹＥＳとなって、図３８ステップ３８０４の処理が終了する。

【０２２７】以上に示されるステップ３８０２～３８０６の処理が図３５の到達可能状態列挙部３５０２により実行されることによって、動作モデルＭにおいて、その初期状態から到達することのできる全ての状態が列挙される。

【０２２８】その後、図３８のステップ３８０７の処理が図３５の試験命令列列挙部３５０３によって実行される。このステップでは、上述の一連の処理によって初期状態から到達することのできる状態の集合Ｕに登録されている、状態ｓと、その状態に至る命令列T_sと、その状態の１時刻前の状態である現在時刻の状態ｓの複数の組合わせを用いて、それら状態によって構成される状態時系列のうち、入力情報保持部３５０１に記憶されている状態時系列の集合内の１つの状態時系列と一致するものが選択されて、その状態時系列を実現する試験命令列３５０４が列挙される。

【０２２９】このステップ３８０７の詳細な動作フローチャートは、図４０に示される。即ち、まず、ステップ４００１では、図３５の入力情報保持部３５０１に入力された状態時系列の集合のうちから１つの状態時系列が選択される。

【０２３０】この選択に成功しステップ４００２の判定がＮＯとなったら、ステップ４００３で、その選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、図３８のステップ３８０６までの処理によって列挙された、動作モデルＭにおいて初期状態から到達することのできる全ての状態の集合Ｕのなかから検索される。

【０２３１】この検索に成功しステップ４００４の判定がＮＯとなったら、ステップ４００５～ステップ４００７の一連の処理の繰返しにより、その検索された状態から順次１時刻（１命令サイクル）ずつバックトラックしながら、その検索された状態を最終状態とする状態時系列が、ステップ４００１によって現在選択されている状態時系列と一致するか否かが検査される。

【０２３２】即ち、まず、ステップ４００５において、ステップ４００３で検索された状態と共に集合Ｕに記憶されているその状態の１時刻前の状態が、ステップ４００１によって現在選択されている状態時系列の最終状態の１時刻前の状態と比較される。

【０２３３】ステップ４００６では、ステップ４００５の比較結果が一致を示しているか否かが判定される。ステップ４００５の比較結果が一致を示しておりステップ４００６の判定がＹＥＳなら、ステップ４００７で、ステップ４００１によって現在選択されている状態時系列において直前に実行されたステップ４００５での比較処理に使用された状態が、その状態時系列の先頭の状態であるか否かが判定される。

【０２３４】ステップ４００７の判定がＮＯならば、ステップ４００５～４００７が再び実行されることにより、ステップ４００３で検索された状態からバックトラックされた状態であって前回の比較処理に使用された状態が集合Ｕ上で検索され、その状態と共に集合Ｕに記憶されているその状態の更に１時刻前の状態が、ステップ４００１によって現在選択されている状態時系列において前回の比較処理に使用された状態の更に１時刻前の状態と比較される。

【０２３５】そして、ステップ４００７の判定がＹＥＳとなるまで、ステップ４００５～ステップ４００６の一連の処理が繰り返されることにより、ステップ４００３によって検索された状態からバックトラックして得られる状態時系列が、ステップ４００１によって現在選択されている状態時系列と一致するか否かが検査されるのである。

【０２３６】以上のようにして、最終的にステップ４００７の判定がＹＥＳとなると、ステップ４００３によって検索された状態からバックトラックして得られる状態時系列が、ステップ４００１によって現在選択されている状態時系列と一致していることになる。

【０２３７】この場合には、ステップ４００８で、ステップ４００３によって検索された最終状態と共に集合Ｕ

に記憶されている、その最終状態に至る命令列が、図３５の試験命令列３５０４として出力される。

【０２３８】その後、ステップ４００３の処理に戻り、ステップ４００１で選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、集合Ｕ内に更に存在するか否かが検索される。これは、図３８のステップ３８０５の説明で前述したように、状態の集合Ｕには、到達経路が異なる複数の同じ到達状態に対して異なるラベルが付与され得るからである。

【０２３９】以上の一連の処理の繰返しにおいて、ステップ４００３で検索された状態からバックトラックされた状態と、ステップ４００１によって現在選択されている状態時系列上で最終状態からバックトラックされた状態とが、途中で一致しなくなると、ステップ４００６の判定がＮＯとなる。この場合には、ステップ４００３で検索された状態からのバックトラックは諦められ、再びステップ４００３の処理に戻って、ステップ４００１で選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、集合Ｕ内に更に存在するか否かが検索される。

【０２４０】また、ステップ４００３の処理の結果、ステップ４００１で選択された状態時系列を構成する複数の状態のうちの最終状態と同じ状態が、集合Ｕ内にそれ以上存在しなくなると又は最初から全く存在しないと、ステップ４００４の判定がＹＥＳとなる。この場合には、ステップ４００１の処理に戻り、図３５の入力情報保持部３５０１に入力された状態時系列の集合のうちから別の１つの状態時系列が選択され、ステップ４００２以降の処理が実行される。

【０２４１】最後に、ステップ４００１の処理の結果、図３５の入力情報保持部３５０１に入力された状態時系列の集合に、これ以上選択されるべき状態時系列が存在しなくなった場合には、ステップ４００２の判定がＹＥＳとなり、図３８のステップ３８０７の処理を終了する。

【０２４２】上述の第３の実施例により、一定期間にわたって特定の状態を発生させるような試験命令列３５０４を、自動的かつ効率的に生成することができる。

＜本発明の第４の実施例＞次に、本発明の第４の実施例について説明する。

【０２４３】第４の実施例では、動作モデルに対して試験項目を指定するための入力情報として、状態時系列の評価関数を指定できる点が特徴である。本発明の第４の実施例の全体構成図は、第２の実施例の場合と同じ図３５によって示される。

【０２４４】第４の実施例においては、入力情報保持部３５０１は、動作モデルＭとそのモデルに対する状態時系列の評価関数とからなる入力情報を保持する。到達可能状態列挙部３５０２は、入力情報保持部３５０１に保持された動作モデルＭにおいて、初期状態から到達することのできる全ての状態を列挙する。

【０２４５】試験命令列列挙部３５０３は、到達可能状態列挙部３５０２によって列挙された、初期状態から到達可能な各状態から、それら各状態によって構成される状態時系列のうち、その状態時系列に対する、入力情報保持部３５０１に記憶されている評価関数の計算結果の値が大きい順に所定個の状態時系列に対応する所定個の最終状態を選択し、各最終状態に至る試験命令列３５０４を列挙する。

【０２４６】以上の構成を有する第４の実施例の動作について、図４１及び図４２の動作フローチャートに従って、詳細に説明する。図４１は、第４の実施例の全体動作フローチャートである。

【０２４７】まず、図４１のステップ４１０１に示されるように、図３５に示される入力情報保持部３５０１には、第１の実施例における図７に示される状態数最小化部７０３から出力される動作モデルＭと予め設定されている状態時系列の評価関数が入力され、それらが保持される。

【０２４８】動作モデルＭは、第１～第３の実施例の場合と同様に、前述した図１６に示される状態変数の構造体の集合として入力情報保持部３５０１に保持される。一方、状態時系列の評価関数は、動作モデルＭが実現する或る任意の期間の状態時系列が入力されるとその評価値を出力する関数であり、動作モデルＭによって示されるパイプライン機構の設計誤りが検出されやすい状態時系列（パイプライン機構に負荷を与える状態時系列）ほど大きな値を出力するように定義される関数である。

【０２４９】具体的には、プログラム上のレジスタ番号と物理レジスタ番号との間で動的に番号の付替えを行う、いわゆるレジスタリネーミング機構を有するパイプラインプロセッサが存在するが、このようなプロセッサにおいては、リネーミングのための物理レジスタの数（リネーミングスペース）が不足する場合が問題となる。この不足が生じないように設計されたプロセッサにおいて本当にこの不足が発生しないことを検証するためには、できるだけ多くのリネーミングスペースが消費されるような試験命令列が必要となる。このような要求を満たす状態時系列の評価関数が、入力情報保持部３５０１に入力され保持される。より具体的には、この例の場合における評価関数は、状態時系列中に含まれる各状態において消費されるリネーミングスペースの、その状態時系列中の全状態にわたる総和を規定するような関数である。

【０２５０】上述したようにして、入力情報保持部３５０１に動作モデルＭと状態時系列の評価関数が保持された後、図３５の到達可能状態列挙部３５０２によって、ステップ４１０２～４１０６の処理が実行される。これらの処理により、動作モデルＭにおいて、その初期状態から到達することのできる全ての状態が列挙される。こ

れらの処理は、第３の実施例における図３８のステップ３８０２～３８０６の一連の処理と同じである。

【０２５１】まず、ステップ４１０２では、第３の実施例における図３８のステップ３８０２と同様に、初期状態から到達することのできる状態の集合Ｕが空集合φとされる。また、状態の集合Ｓに、初期状態が設定される。

【０２５２】続いて、ステップ４１０３～４１０６の一連の処理が繰り返し実行される。まず、ステップ４１０３では、第３の実施例における図３８のステップ３８０３と同様に、動作モデルＭにおいて、現在の時刻（命令サイクル）までに到達している状態の集合Ｕに、１時刻（１命令サイクル）前に到達した状態の集合Ｓの内容が追加される。なお、状態の集合Ｓの初期値は、ステップ４１０２で設定される初期状態の集合である。

【０２５３】ステップ４１０４においては、第３の実施例における図３８のステップ３８０４と同様に、動作モデルＭにおいて、１時刻（１命令サイクル）前に到達した状態の集合Ｓに含まれる各状態から、基本遷移により到達する次時刻（次の命令サイクル）の各状態が、次時刻の状態の集合Ｓ′として計算される。この処理の詳細は、第３の実施例において示した図３９の動作フローチャートによって示される。

【０２５４】ステップ４１０５においては、第３の実施例における図３８のステップ３８０５と同様に、次時刻の状態の集合Ｓ′に含まれる状態が、次時刻に到達した状態の集合Ｓとされる。この場合、第３の実施例の場合と同様に、次時刻の状態の集合Ｓ′に含まれる状態が、現在の時刻までに到達している状態の集合Ｕに含まれていても、その状態は次時刻に到達した状態の集合Ｓとされる。これは、第４の実施例においても、ある１時刻における状態だけではなく状態の時系列が問題となるため、状態の遷移順序を保存する必要があるためである。従って、状態の集合Ｕには、複数の同じ到達状態に対して異なるラベルが付与され得る。

【０２５５】また、ステップ４１０５では、第３の実施例における図３８のステップ３８０５と同様に、次時刻の状態の集合Ｓ′に含まれる状態のうち現在の時刻までに到達している状態の集合Ｕに含まれない状態も状態Ｓ″として保存される。

【０２５６】そして、続くステップ４１０６で、第３の実施例における図３８のステップ３８０６と同様に、この状態Ｓ″が空集合φであるか否かが判定される。即ち、どのように状態が遷移しても新しい状態が１つも発生しなくなりこのステップ４１０６の判定がＹＥＳとなった時点で、到達可能状態列挙部３５０２の動作が終了することになる。

【０２５７】ステップ４１０６の判定がＮＯならば、次時刻に到達した状態の集合Ｓから更に次時刻における基本遷移によって到達できる状態を検索するために、ステップ４１０３～４１０６の一連の処理が繰り返される。

【０２５８】以上の図４１のステップ４１０３～４１０６の一連の処理の繰返しによって、例えば図３０に示される初期状態からの基本遷移のみによって到達できる全ての状態が列挙される。

【０２５９】その後、図４１のステップ４１０７と４１０８の処理が図３５の試験命令列列挙部３５０３によって実行される。まずステップ４１０７では、上述の一連の処理によって初期状態から到達することのできる状態の集合Ｕに登録されている、状態ｓと、その状態に至る命令列T_sと、その状態の１時刻前の状態である現在時刻の状態ｓの複数の組合わせを用いて、それら状態によって構成される全ての状態時系列について、その状態時系列の評価関数が計算される。

【０２６０】このステップ４１０７の詳細な動作フローチャートは、図４２に示される。即ち、まず、ステップ４２０１では、図４１のステップ４１０６までの処理によって列挙された、動作モデルＭにおいて初期状態から到達することのできる全ての状態の集合Ｕのなかから、１つの到達状態が選択される。

【０２６１】この検索に成功しステップ４２０２の判定がＮＯとなったら、ステップ４２０３で、ステップ４２０１で選択された状態を用いて、現在バックトラックされている状態時系列の評価関数が計算される。具体的には、その状態におけるリネーミングスペースが計算され、現在バックトラックされている状態時系列の評価関数の計算結果に累算される。上述のリネーミングスペースの計算は、第１の実施例の場合と同様の例えば図３１に示されるようなデータ形式を有する１つの状態について、各状態変数に対応する例えば図１６及び図１７に示されるデータ形式を有する動作テーブルに設定されているリネーミングスペースを、当該状態内の全状態変数にわたって合計する計算として実現できる。

【０２６２】その後、ステップ４２０４～４２０６の一連の処理の繰返しにより、ステップ４２０１で選択された状態から順次１時刻（１命令サイクル）ずつバックトラックしながら、初期状態に到達するまで、バックトラックによって生成される状態時系列に対する評価関数が順次計算される。

【０２６３】即ち、まず、ステップ４２０４において、ステップ４２０１で選択された状態と共に集合Ｕに記憶されているその状態の１時刻前の状態が選択される。次に、ステップ４２０５において、ステップ４２０３の場合と同様に、ステップ４２０４で選択された状態を用いて、現在バックトラックされている状態時系列の評価関数が計算される。即ち、ステップ４２０４で選択された状態におけるリネーミングスペースが計算され、現在バックトラックされている状態時系列の評価関数の計算結果に累算される。

【０２６４】ステップ４２０６では、ステップ４２０４

で選択された状態が動作モデルMにおける初期状態であるか否かが判定される。ステップ４２０６の判定がＮＯなら、ステップ４２０４の処理に戻り、状態時系列のバックトラックが進められる。

【０２６５】ステップ４２０６の判定がＹＥＳとなると、ステップ４２０１で選択された状態を最終状態とする状態時系列のバックトラックが終了し、その状態時系列に対する最終的な評価関数の計算結果が算出されたことになる。

【０２６６】この後、ステップ４２０７で、集合U内の、ステップ４２０１で選択された状態に対応する要素位置に、その状態を最終状態とする状態時系列に対する上述の評価関数の計算結果が登録される。

【０２６７】その後、ステップ４２０１の処理に戻り、集合Uのなかから別の１つの到達状態が選択され、ステップ４２０２以降の処理が繰り返される。そして、集合Uにおいてこれ以上選択されるべき到達状態が存在しなくなり、ステップ４２０２の判定がＹＥＳとなると、図４１のステップ４１０７の処理を終了する。

【０２６８】最後に、図４１のステップ４１０８では、上述の一連の処理によって初期状態から到達することのできる状態の集合Uに登録されている、各到達状態（各状態時系列の最終状態）と、その到達状態に至る命令列と、その到達状態を最終状態とする状態時系列の評価関数の計算結果の組合せにおいて、評価関数の値が大きい順に、所定個の到達状態が選択され、その到達状態に至る命令列が上記集合Uから取り出され、それが試験命令列３５０４（図３５）として出力される。

【０２６９】上述の第４の実施例により、一定期間にわたって定性的に表現される試験状態となる試験命令列３５０４を、自動的かつ効率的に生成することができる。

＜その他の実施態様＞最後に、本発明のその他の実施態様について説明する。

【０２７０】まず、本発明では、自動生成された試験命令列に、プロセッサの初期状態を設定する命令列及び終了状態を保存する命令列が付加されることにより、パイプライン制御機構の検証用の試験命令列を生成することもできる。

【０２７１】また、本発明は、試験命令列の自動生成のみならず、プロセッサを構成するパイプラインに対応する動作モデルに対して何等かの解析を行う一般的な技術に適用できる。特に、動作モデルに対して状態数最小化処理を実行する技術を用いることにより、その処理の結果として得られる動作モデルに対応して、最適化されたパイプラインを再構成することも可能となる。

【０２７２】

【発明の効果】本発明の第１～第３の態様によれば、パイプラインに関する基本的な情報を入力するだけで、自動的にプロセッサの動作モデルや試験命令列（テストプログラム）を生成することが可能となる。この結果、テストプログラムの開発者は、初期設計及び設計変更に伴う工数を大幅に削減できるため、プロセッサの開発サイクル短縮に寄与するところが大きい。

【０２７３】特に、本発明の第２の態様により構成される動作モデルは、試験命令列の自動生成の効率化を実現するのみならず、構成される動作モデルに対応する最適化されたパイプラインを再構成することも可能となる。この結果、本発明は、初期的なハードウエア設計のための技術としても利用することができる。

【０２７４】また、本発明の第３の態様によれば、試験命令列３０２の抽出の処理を大幅に効率化することが可能となる。本発明の第４の態様によれば、トラップ制御などの定性的に表現される試験項目を、状態を評価する状態評価関数によって効率的に定義することができ、それに対応する試験命令列を、自動的かつ効率的に生成することが可能となる。

【０２７５】本発明の第５の態様によれば、一定期間命令の完了順序が入れ替わるような状態の時系列を状態の時系列として指定することができ、一定期間にわたって特定の状態を発生させるような試験命令列を、自動的かつ効率的に生成することが可能となる。

【０２７６】本発明の第６の態様では、一定期間の状態が定性的に表現される試験項目を、状態の時系列を評価する評価関数によって効率的に定義することができ、それに対応する試験命令列を、自動的かつ効率的に生成することが可能となる。

【０２７７】更に、本発明の全般について挙げることができる効果として、本発明により自動生成された試験命令列の集合は、それぞれ異なるパイプラインハザード発生ケースに対応させることができるのみならず、それぞれの集合から代表的な要素が取り出されることによって、様々なパイプラインハザードの発生ケースを網羅的に検証することが可能となる。

【０２７８】また、現在のプロセッサにおいては、命令の種類によりパイプライン経路が変更されたりレジスタ値がフォワーディング等されるため、パイプラインがインターロックを起こさない場合においても制御機構の検証が重要となる場合がある。本発明により自動生成される試験命令列の集合は、それぞれが特定のパイプライン制御パターンに対応しているため、それら制御パターンに対応して特定の制御機構が動作する状況を列挙するような作業にも役立てることができる。

【図面の簡単な説明】

【図１】本発明のブロック図（その１）である。

【図２】本発明のブロック図（その２）である。

【図３】本発明のブロック図（その３）である。

【図４】本発明のブロック図（その４）である。

【図５】本発明のブロック図（その５）である。

【図６】本発明のブロック図（その６）である。

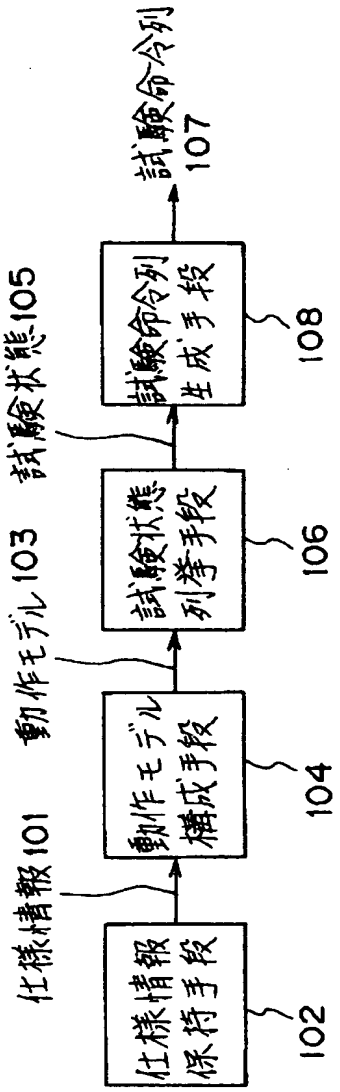
【図７】本発明の第１の実施例の全体構成図である。

【図８】第１の実施例の全体動作フローチャート（その１）である。
【図９】状態数最小化処理の動作フローチャートである。
【図１０】入力段側からの状態数最小化処理の動作フローチャートである。
【図１１】最終段側からの状態数最小化処理の動作フローチャートである。
【図１２】第１の実施例の全体動作フローチャート（その２）である。
【図１３】試験命令列列挙処理の動作フローチャートである。
【図１４】基本遷移による次時刻の状態集合Ｓ′の計算と試験命令列出力の処理の動作フローチャートである。
【図１５】次時刻状態集合の計算処理の動作フローチャートである。
【図１６】状態変数の構造体を示した図である。
【図１７】動作テーブルのデータ構造を示した図である。
【図１８】仕様情報によって表されるパイプライン構成の例を示した図である。
【図１９】仕様情報によって表される命令の実行形態の例を示した図である。
【図２０】初期動作モデルM_0の例を示した図である。
【図２１】動作モデルM_1の例を示した図である。
【図２２】動作モデルM_2の例を示した図である。
【図２３】試験状態の集合Ｈの例を示した図である。
【図２４】初期状態からの基本遷移のみによって到達できるケース１の試験状態を発生させる試験命令列の例を示した図である。
【図２５】ケース１－１、１－２、１－３、１－４の状態の次時刻の状態を示した図である。
【図２６】ケース１とケース１′の関係を示した図である。
【図２７】ケース１′－１、１′－２、１′－３、１′－４の状態からの基本遷移のみによえ到達できるケース５の試験状態を発生させる試験命令列の例を示した図である。
【図２８】ケース１′とケース５の関係を示した図である。
【図２９】試験状態のデータ構造の例を示した図である。
【図３０】初期状態のデータ構造の例を示した図である。
【図３１】基本遷移の後のｓ′とT_sのデータ構造の例（その１）を示した図である。
【図３２】基本遷移の後のｓ′とT_sのデータ構造の例（その２）を示した図である。
【図３３】基本遷移の後のｓ′とT_sのデータ構造の例（その３）を示した図である。
【図３４】試験状態に対応するｓ′とT_sのデータ構造の例を示した図である。
【図３５】本発明の第２～第４の実施例の全体構成図である。
【図３６】第２の実施例の全体動作フローチャートである。
【図３７】第２の実施例における基本遷移による次時刻の状態集合Ｓ′の計算の処理の動作フローチャートである。
【図３８】第３の実施例の動作フローチャートである。
【図３９】第３及び第４の実施例における基本遷移による次時刻の状態集合Ｓ′の計算の処理の動作フローチャートである。
【図４０】第３の実施例における実現可能な状態時系列の計算と試験命令列ん出力の処理の動作フローチャートである。
【図４１】第４の実施例の動作フローチャートである。
【図４２】第４の実施例における状態時系列の選択とその評価関数の計算の処理の動作フローチャートである。
【符号の説明】
１０１、２０１　　仕様情報
１０２、２０２　　仕様情報保持手段
１０３、２０５、３０１　　動作モデル
１０４　　動作モデル構成手段
１０５、３０２　　試験状態
１０６　　試験状態列挙手段
１０７、３０５　　試験命令列
１０８　　試験命令列生成手段
２０３　　初期動作モデル
２０４　　初期動作モデル構成手段
２０６　　状態数最小化手段
３０３　　入力情報保持手段
３０４　　入力状態
３０６　　試験命令列列挙手段
３０７　　次時刻状態
３０８　　次時刻状態計算手段
４０１、４０７、５０１、５０７、６０１、６０７　動作モデル
４０２、４０８　　状態評価関数
４０３、５０３、６０３　　入力情報保持手段４０３
４０４　　到達可能状態列挙手段
４０５、４１３、５０５、５１３、６０５、６１３　試験命令列
４０６、５０６、６０６　　試験命令列列挙手段
４０９、５０９、６０９　　入力情報保持手段
４１０、５１０、６１０　　到達可能状態列挙手段
４１１　　状態評価関数計算手段
４１２、５１１、６１１　　命令列列挙手段
４１４、５１４、６１４　　試験命令列出力手段
５０２、５０８　　状態の時系列

６０２、６０８　　状態の時系列の評価関数
５０４、６０４　　到達可能状態時系列列挙手段
７０１　　仕様情報保持部
７０２　　初期動作モデル構成部
７０３　　状態数最小化部
７０４、３５０１　　入力情報保持部
７０５　　試験命令列列挙部
７０６　　次時刻状態計算部
７０７、３５０４　　試験命令列
３５０２　　到達可能状態列挙部
３５０３　　試験命令列列挙部
M_0　　初期動作モデル
M_1、M_2、M　　動作モデル
H　　試験状態の集合

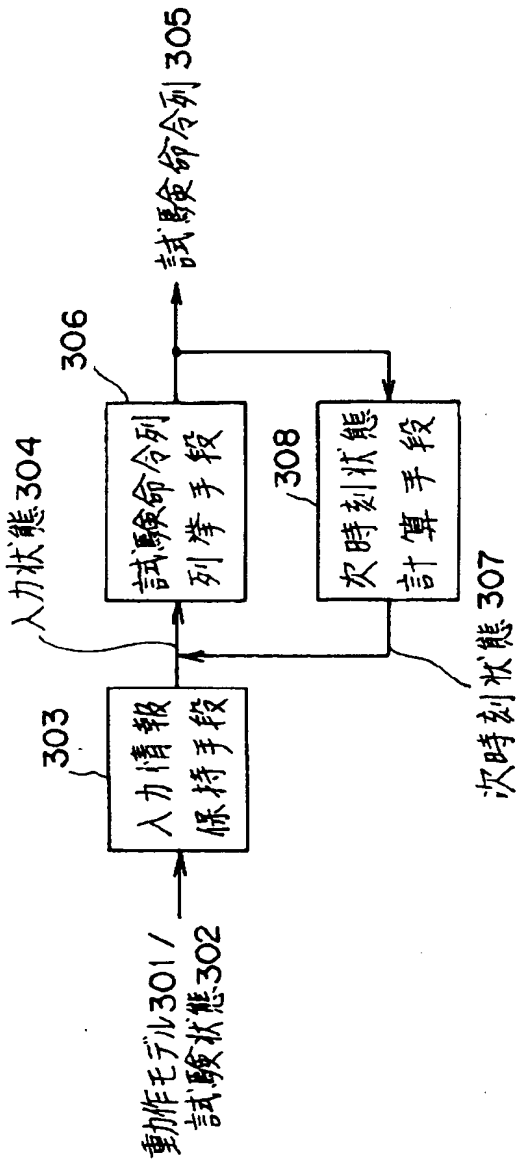
【図１】

本発明のブロック図（その１）



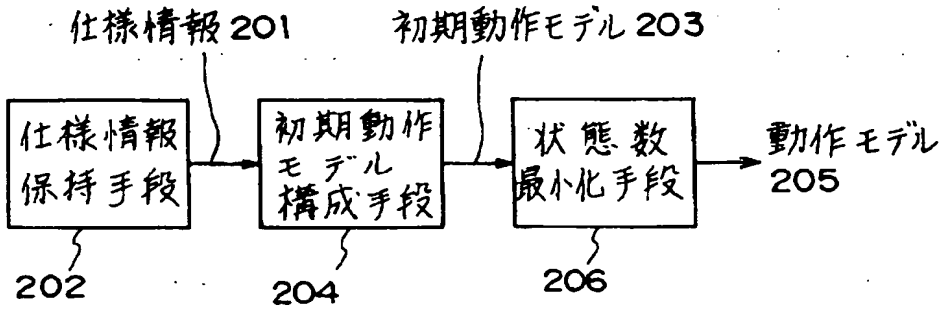
【図３】

本発明のブロック図（その３）



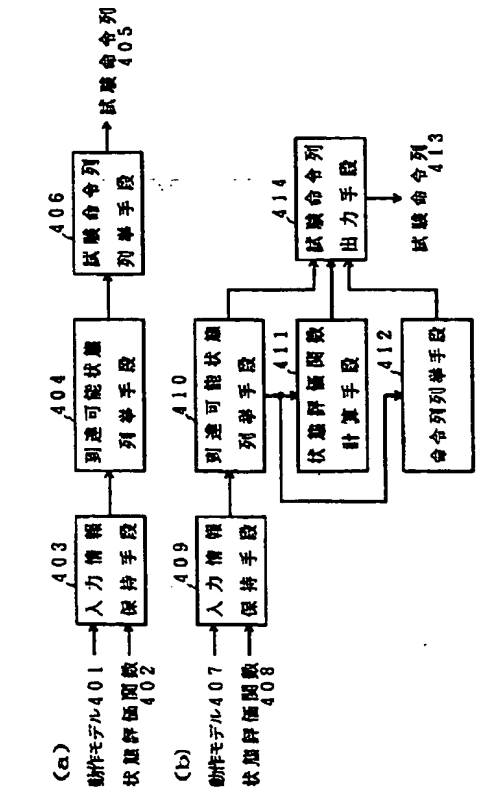
【図２】

本発明のブロック図（その２）



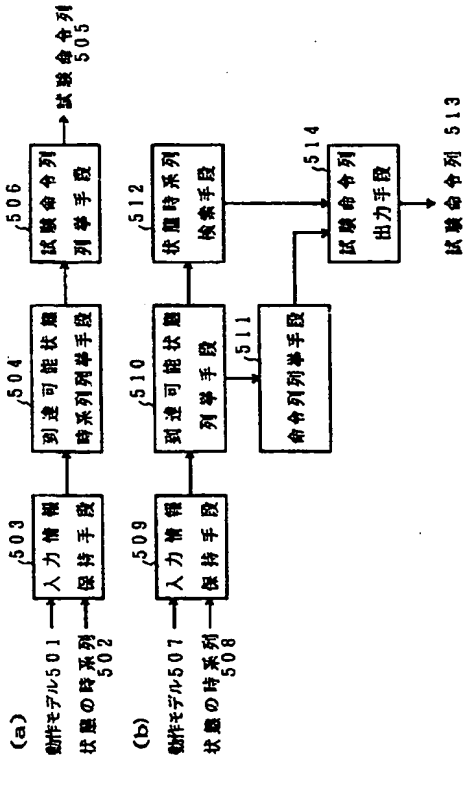
【図４】

本発明のブロック図（その４）



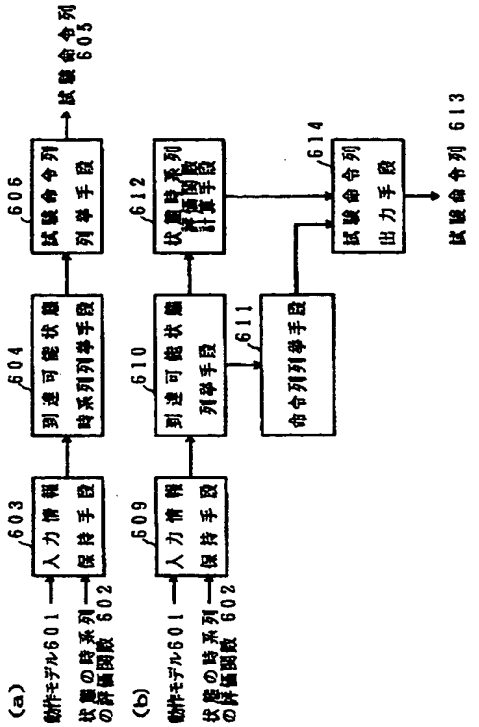
【図５】

本発明のブロック図（その５）



【図６】

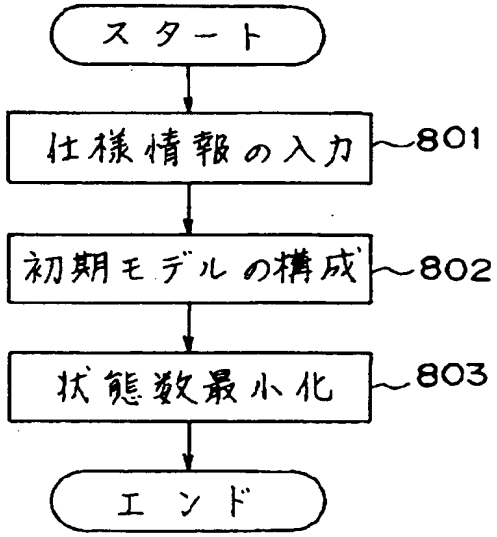
本発明のブロック図（その６）



【図１６】

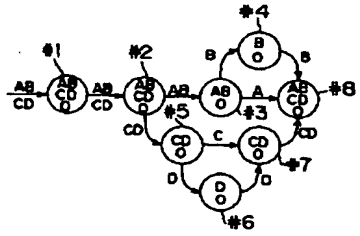
【図８】

第1の実施例の全体動作フローチャート（その1）



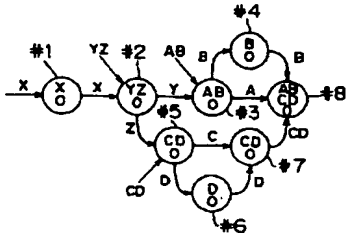
【図２０】

初期動作モデルMoの例を示した図



【図２１】

動作モデルM1を示した図



【図１７】

状態変数の構造体を示した図

前段に接続する状態変数のリスト
後段に接続する状態変数のリスト
入力の集合
状態の集合
遷移関数
動作テーブル

【図１９】

動作テーブルのデータ構造を示した図

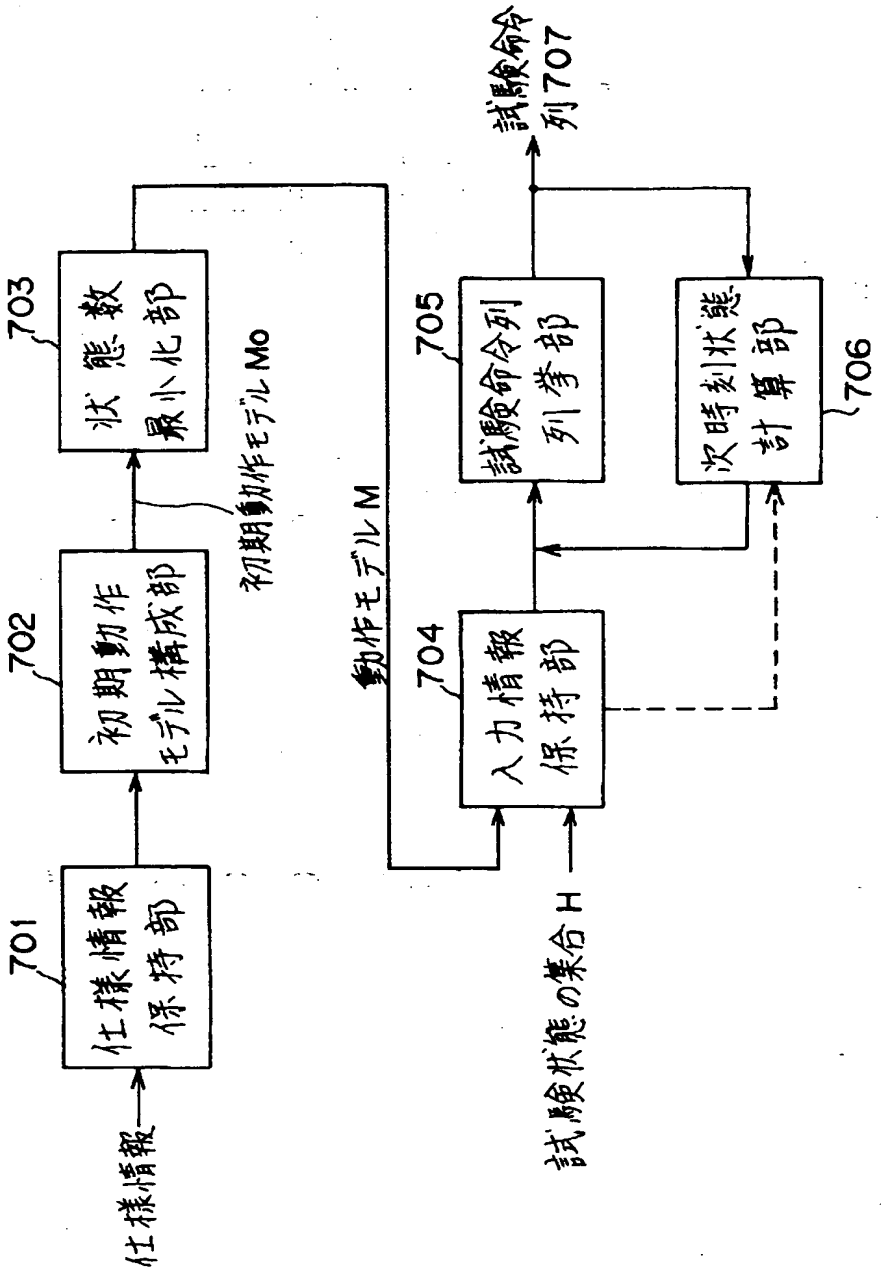
	動作1	動作2	動作3	…
状態1	✓	✓		
状態2	✓	✓		
状態3	✓		✓	
⋮				

仕様情報によって表される命令の実行形態の例を示した図

命令A： IF — ID — ALU — WB
命令B： IF — ID — ALU — MEM — WB
命令C： IF — ID — F1 — F3 — WB
命令D： IF — ID — F1 — F2 — F3 — WB

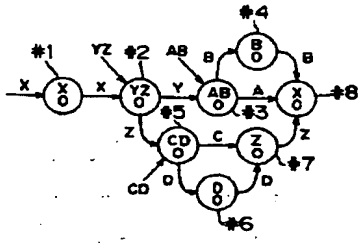
【図７】

本発明の第１の実施例の全体構成図



【図２２】

動作モデルM2を示した図



【図２９】

試験状態のデータ構造の例を示した図

ケース1

状態変数	状態
#1	
#2	
#3	
#4	
#5	C
#6	D
#7	
#8	

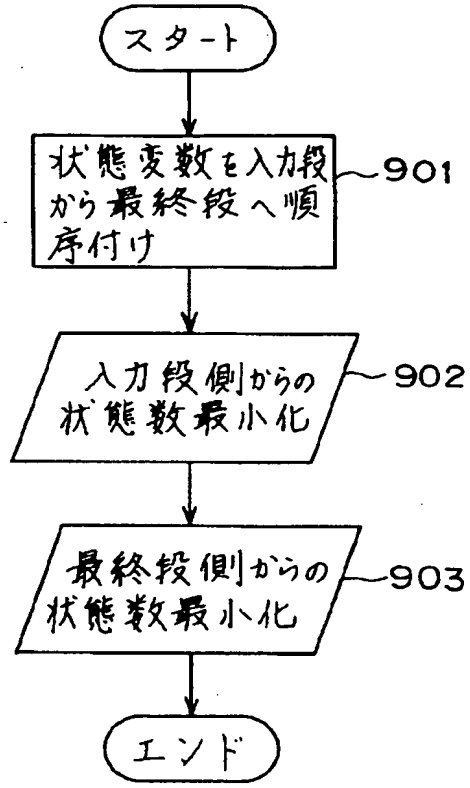
【図３０】

初期状態のデータ構造の例を示した図

状態変数	状態
#1	0
#2	0
#3	0
#4	0
#5	0
#6	0
#7	0
#8	0

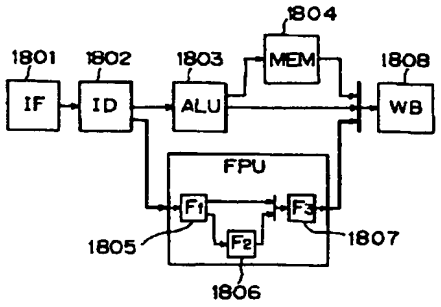
【図９】

状態数最小化処理の動作フローチャート



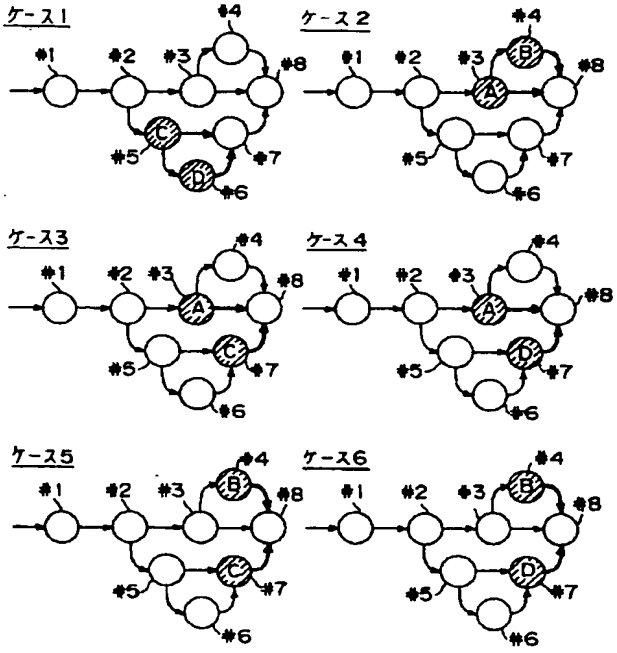
【図１８】

仕様情報によって表される
パイプライン構成の例を示した図



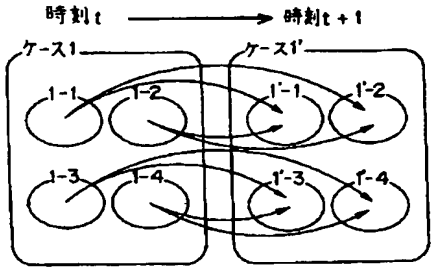
【図２３】

試験状態の集合Hの例を示した図



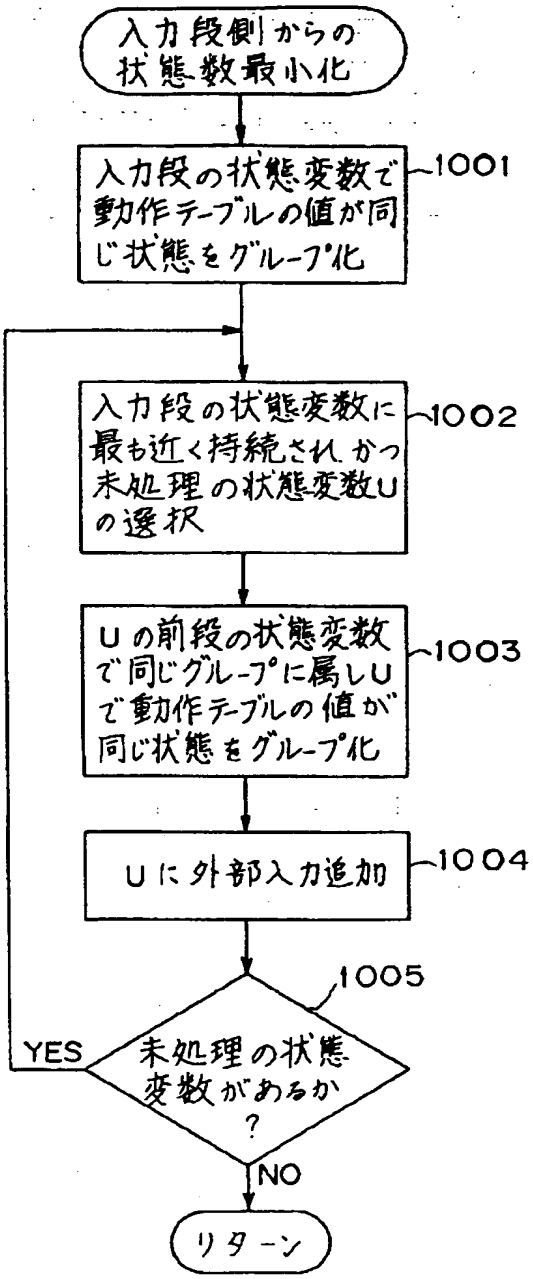
【図２６】

ケース１とケース１'の関係を示した図



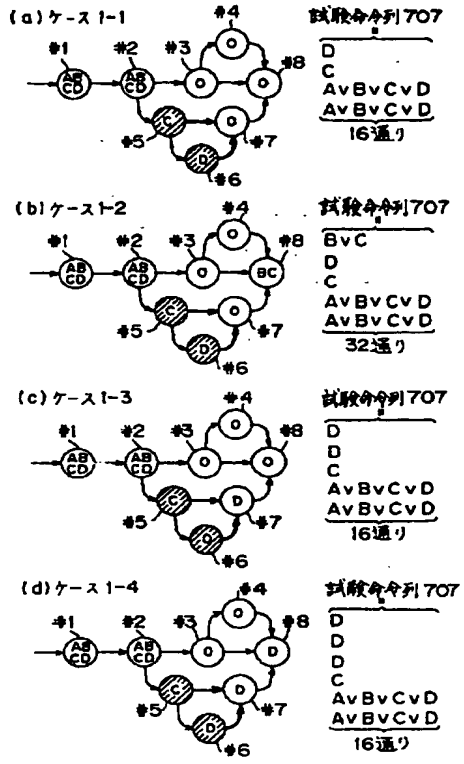
【図１０】

入力段側からの状態数最小化処理の動作フローチャート



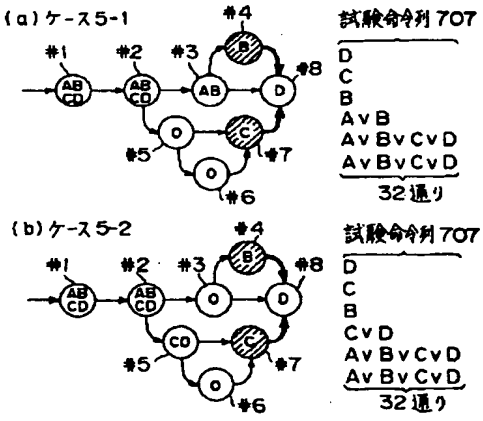
【図２４】

初期状態からの基本遷移のみによって到達できるケース1の試験状態を発生させる試験命令列の例を示した図



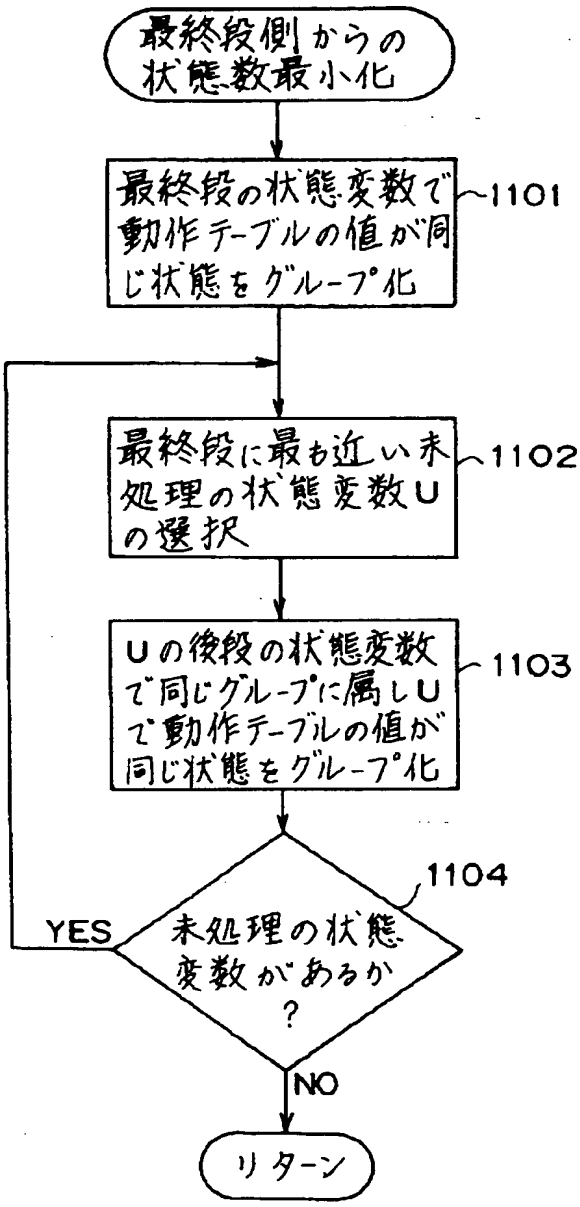
【図２７】

ケース1'-1, 1'-2, 1'-3, 1'-4の状態からの基本遷移のみによって到達できるケース5の試験状態を発生させる試験命令列の例を示した図



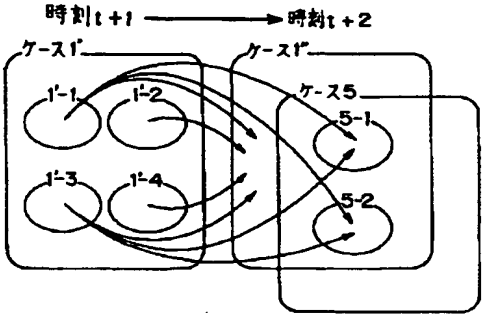
【図１１】

最終段側からの状態数最小化処理の動作フローチャート



【図２８】

ケース1'とケース5の関係を示した図



【図３１】

基本遷移の後のS'とT.'のデータ構造の例（その１）を示した図

(a)

状態変数	状態
#1	A
#2	O
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

S' :　T.' = {A}

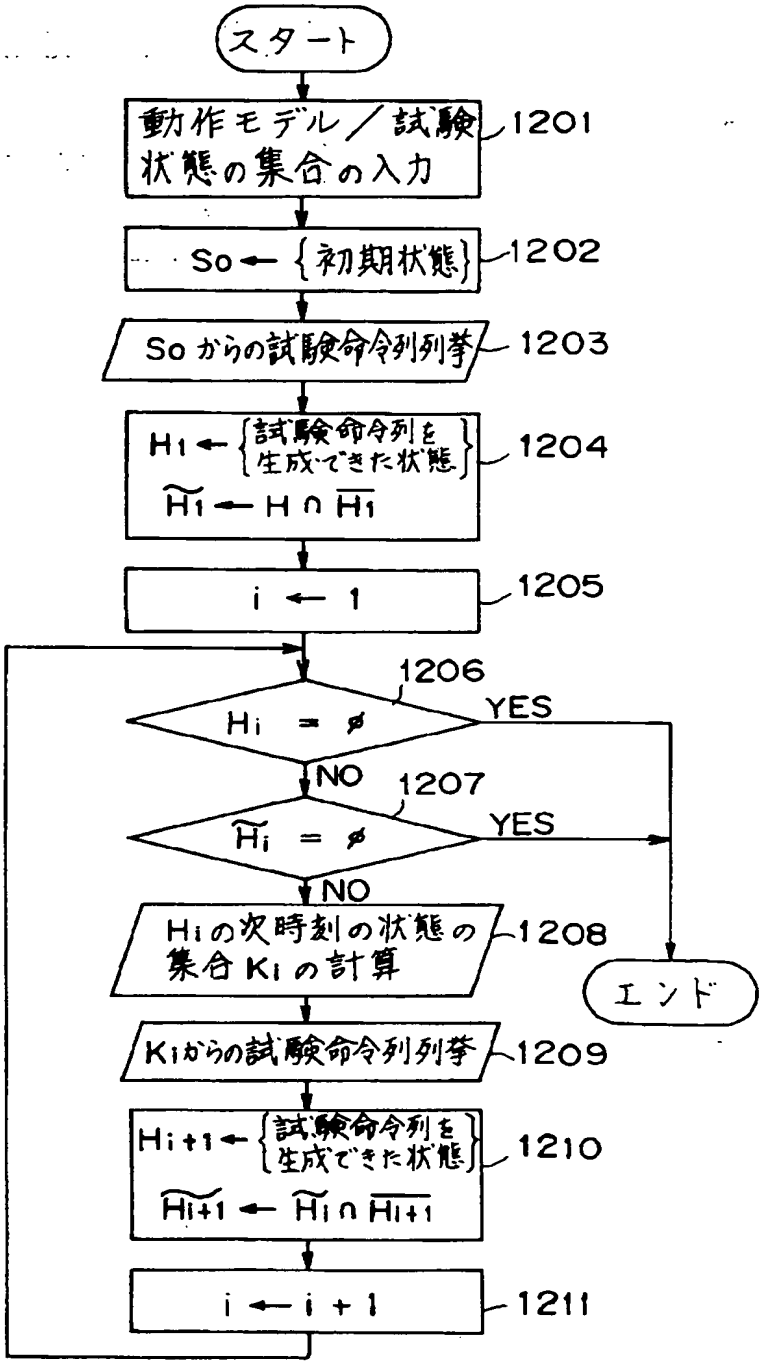
(b)

状態変数	状態
#1	B
#2	O
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

S' :　T.' = {B}

【図１２】

第1の実施例の全体動作フローチャート（その2）



【図３４】

試験状態に対応するS'とT。'のデータ構造の例を示した図

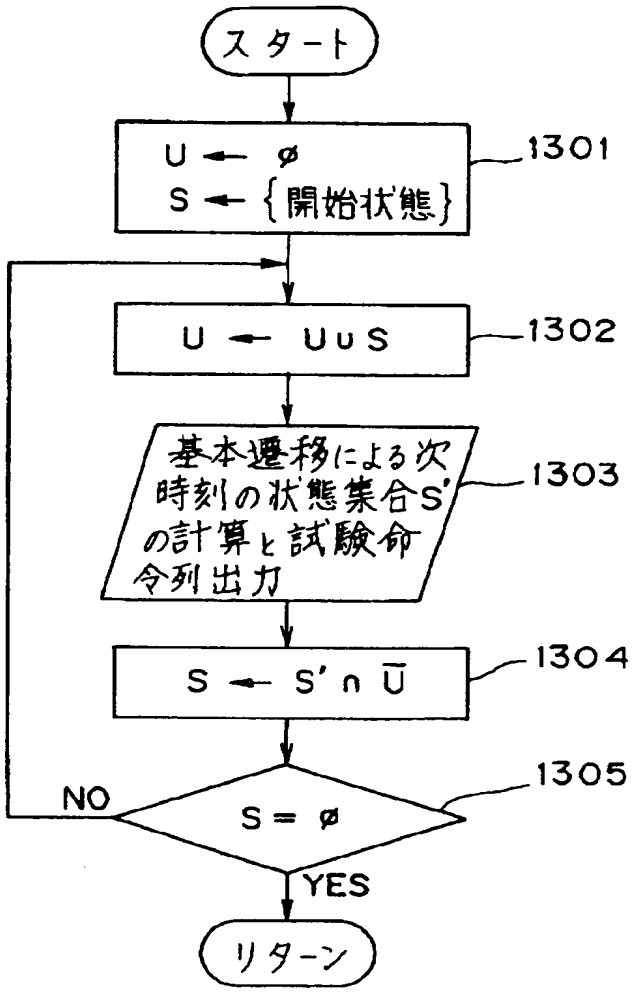
S' :

状態変数	状態
#1	A
#2	A
#3	O
#4	O
#5	C
#6	D
#7	O
#8	O

T。' = {D, C, A, A}

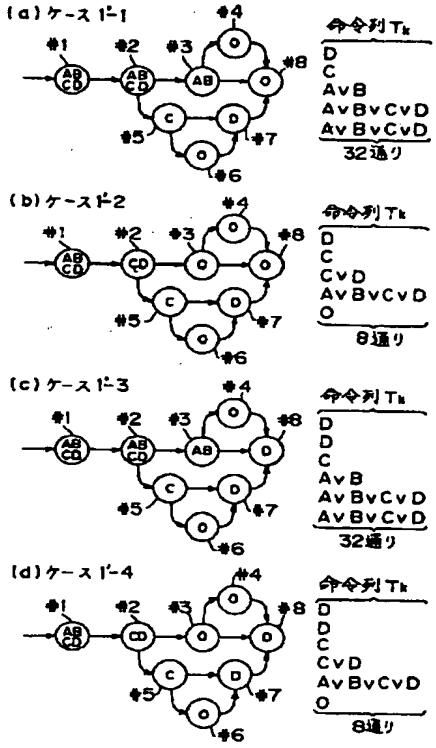
【図１３】

試験命令列列挙処理の動作フローチャート



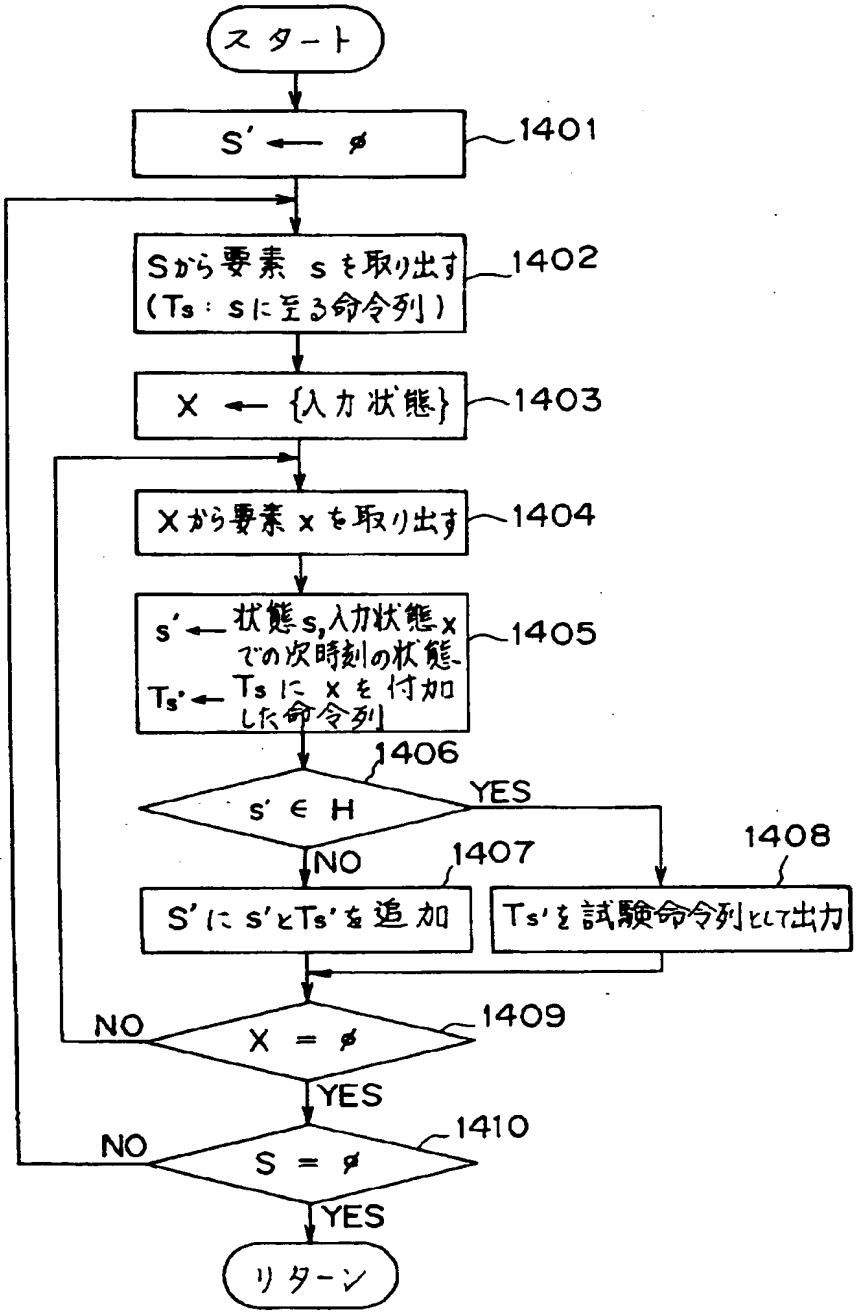
【図２５】

ケース1-1,1-2,1-3,1-4の状態の次時刻の状態を示した図



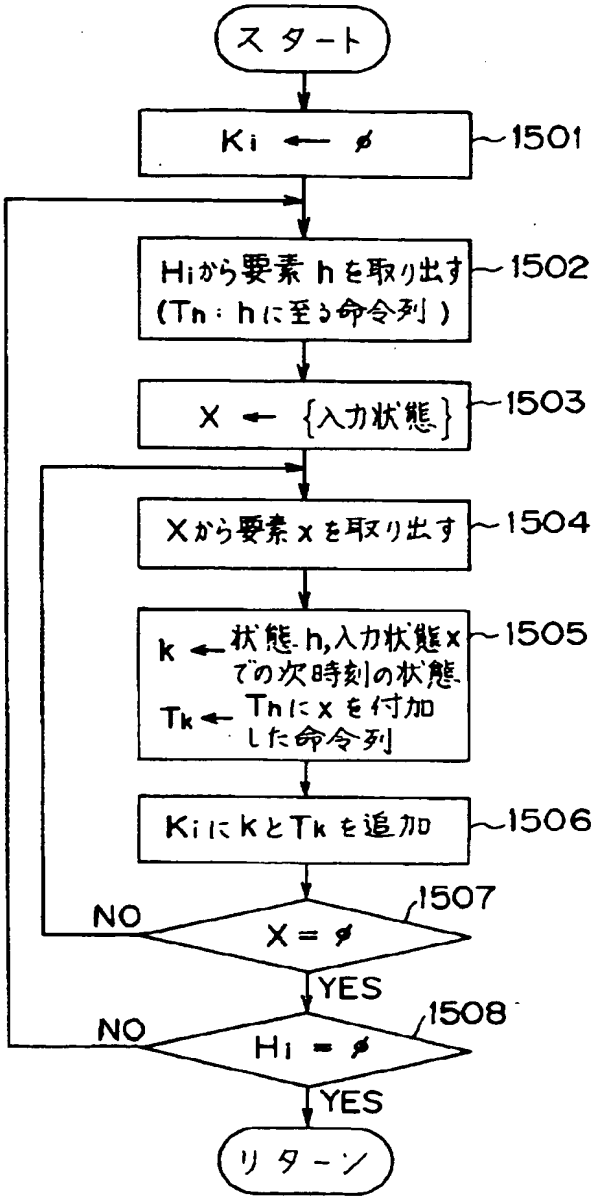
【図１４】

基本遷移による次時刻の状態集合S'の計算と
試験命令列出力の処理の動作フローチャート



【図15】

次時刻状態集合の計算処理の動作フローチャート



【図３２】

基本遷移の後のS′とT。′のデータ構造の例（その２）を示した図

(a)

S′ :

状態変数	状態
#1	A
#2	A
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

T。′ = {A, A}

(b)

S′ :

状態変数	状態
#1	B
#2	A
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

T。′ = {A, B}

【図３３】

基本遷移の後のS′とT。′のデータ構造の例（その３）を示した図

(a)

S′ :

状態変数	状態
#1	A
#2	B
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

T。′ = {B, A}

(b)

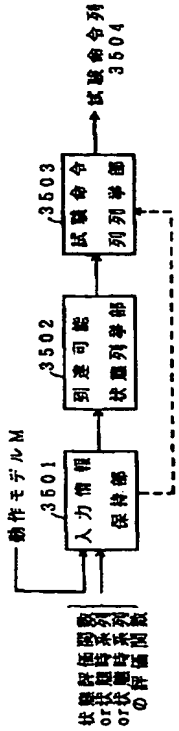
S′ :

状態変数	状態
#1	B
#2	B
#3	O
#4	O
#5	O
#6	O
#7	O
#8	O

T。′ = {B, B}

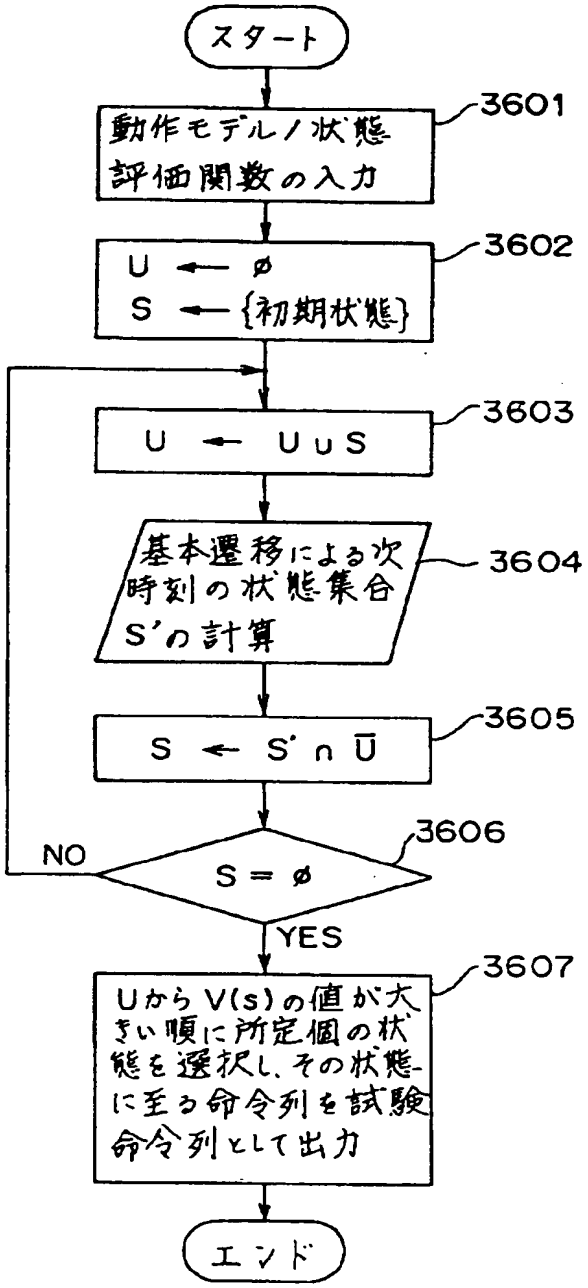
【図35】

本発明の第2～第4の実施例の全体構成図



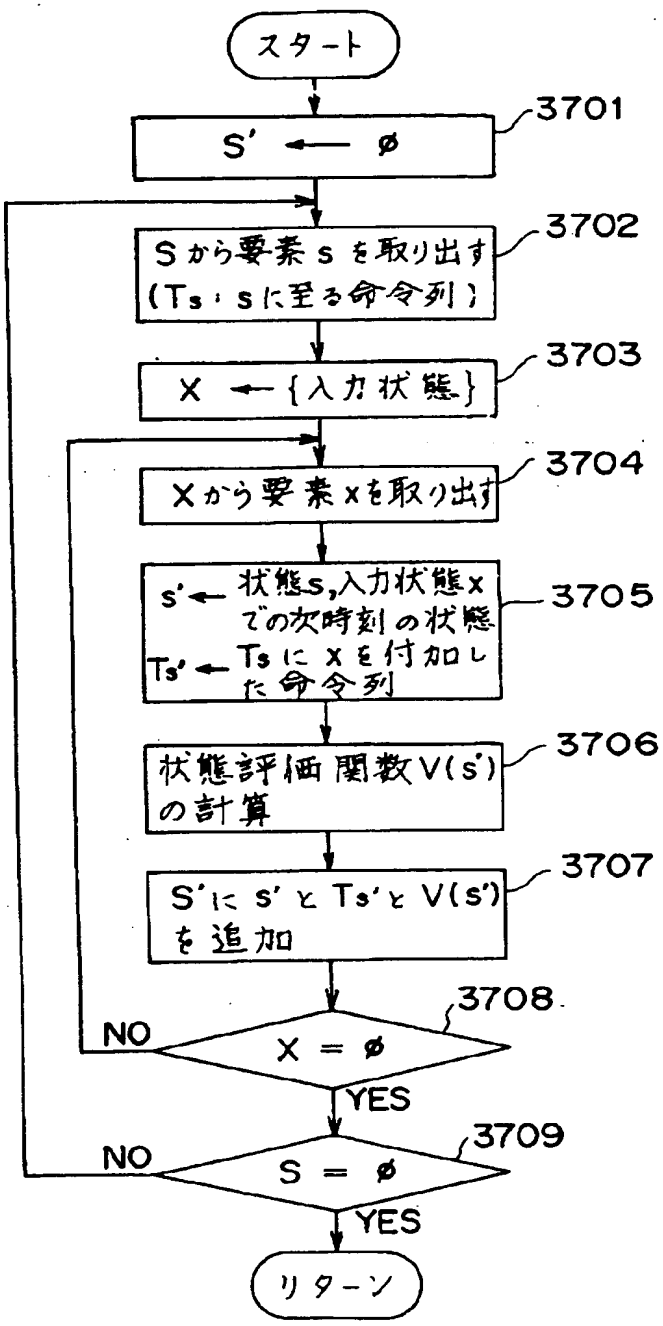
【図36】

第2の実施例の全体動作フローチャート



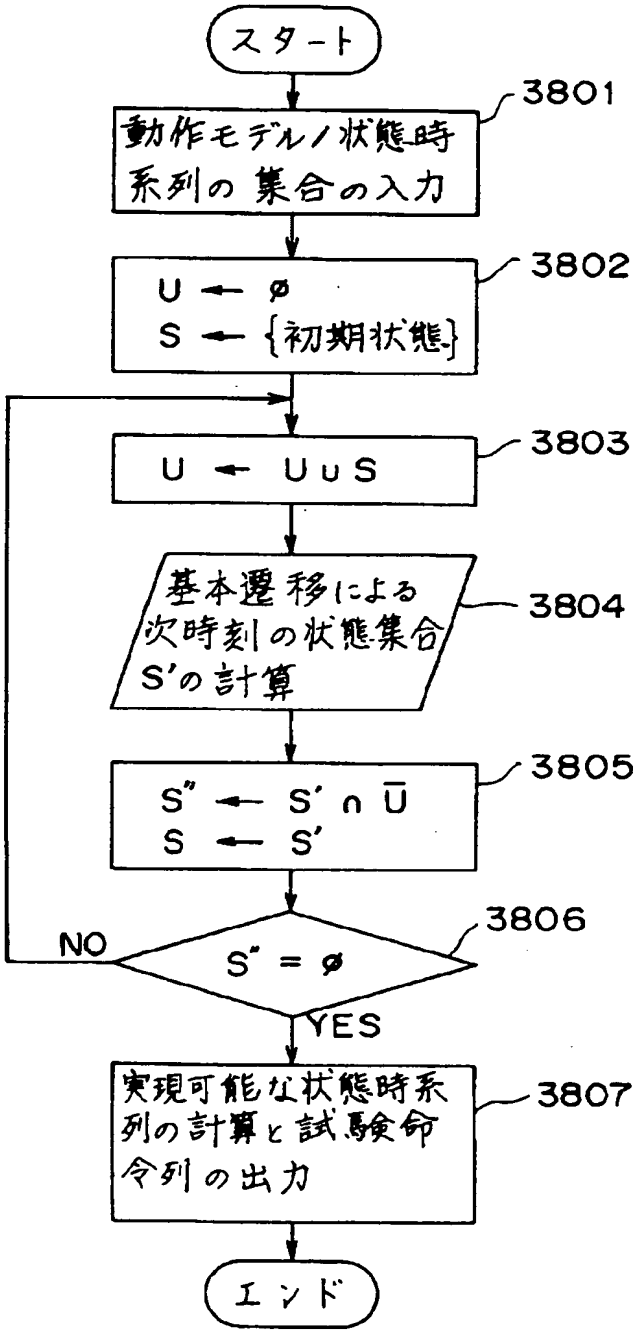
【図３７】

第２の実施例における基本遷移による次時刻の状態集合S'の計算の処理の動作フローチャート



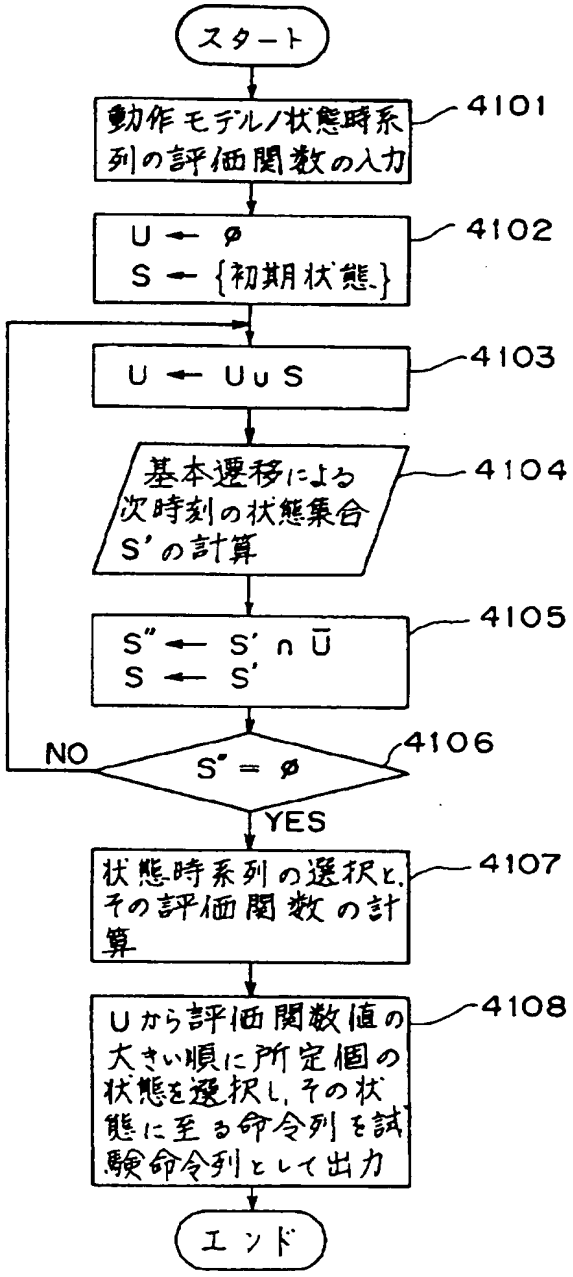
【図３８】

第３の実施例の動作フローチャート



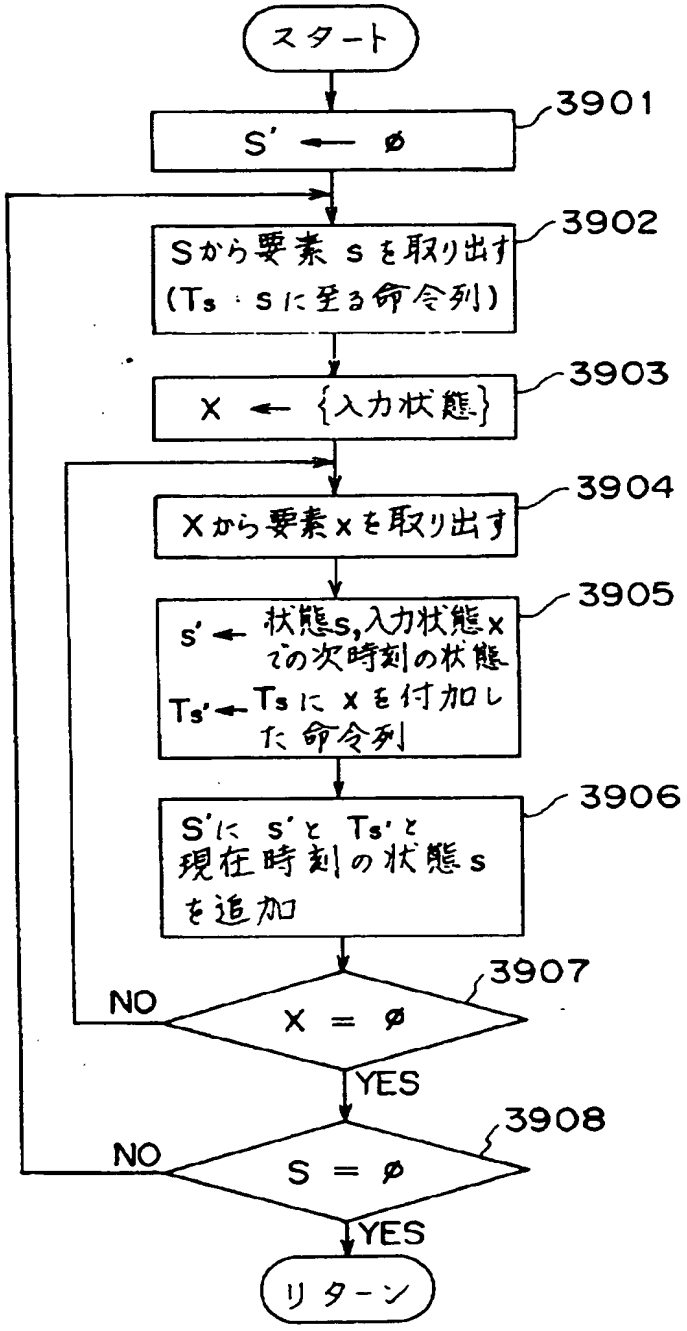
【図４１】

第４の実施例の動作フローチャート



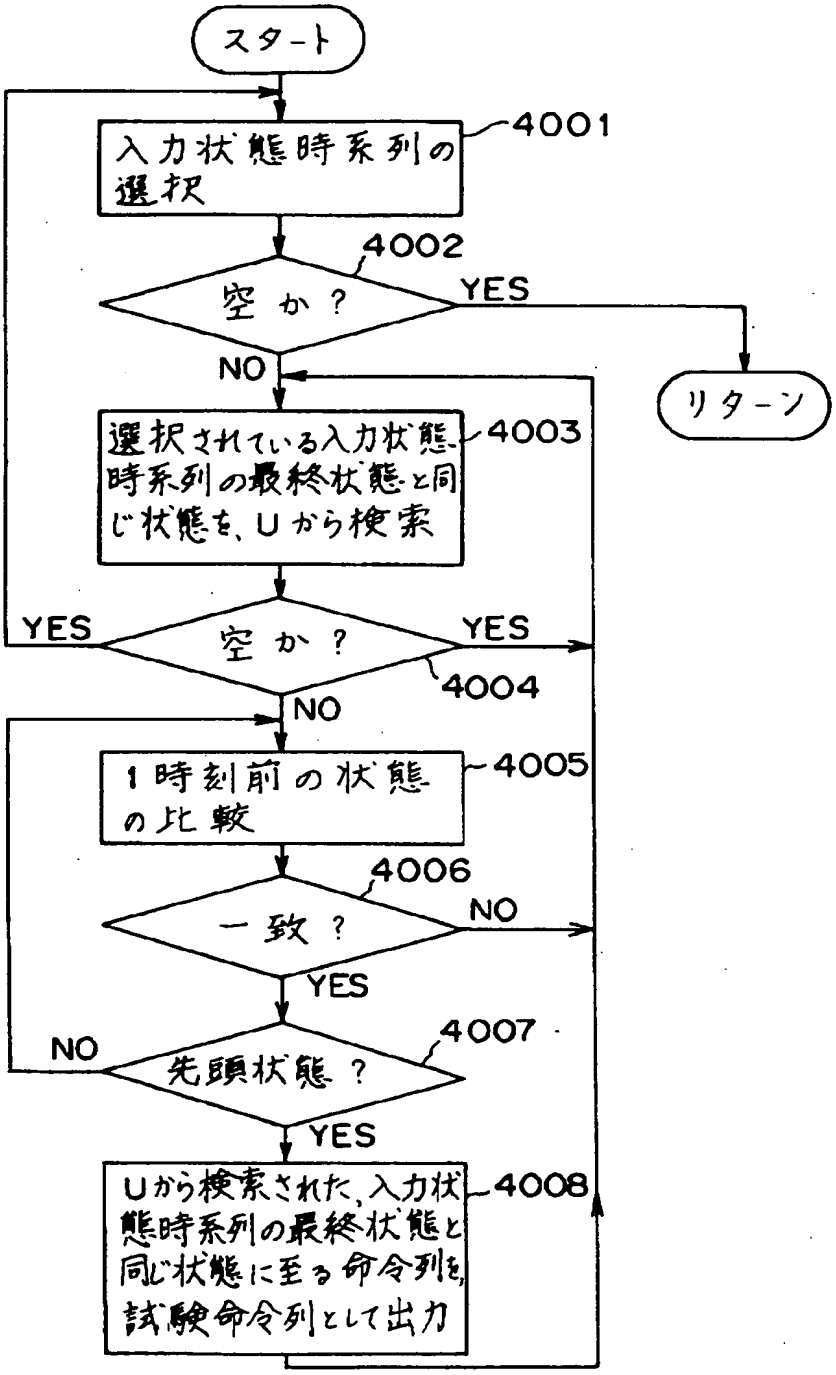
【図３９】

第３及び第４の実施例における基本遷移による次時刻の状態集合S'の計算の処理の動作フローチャート



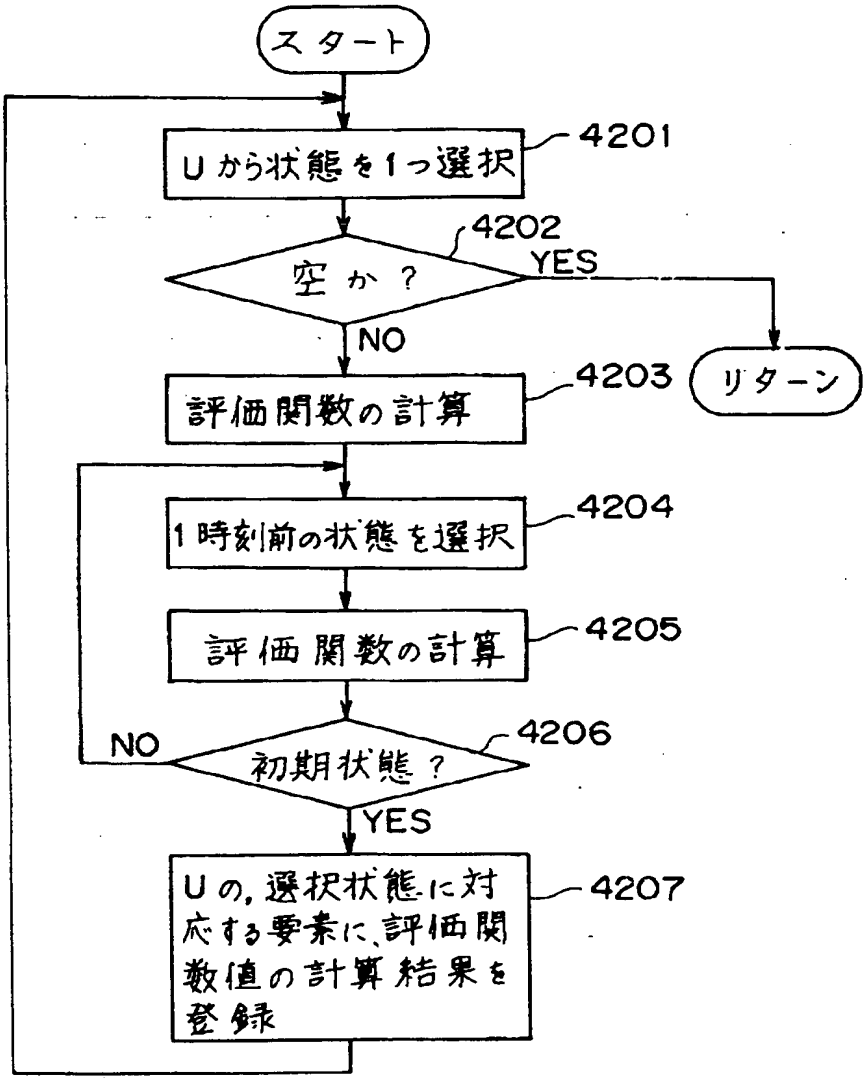
【図40】

第3の実施例における実現可能な状態時系列の計算と試験命令列の出力の処理の動作フローチャート



【図４２】

第４の実施例における状態時系列の選択と
その評価関数の計算の処理の動作フローチャート



フロントページの続き

(72) 発明者　広瀬　文保
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内